CHAPTER 3 DIGITAL SPECTRAL ANALYSIS
CHAPTER 3 DIGITAL SPECTRAL ANALYSIS

3.1 Introduction

The transformation of data from the time domain to the frequency was introduced in Ch. 2. The principles and practice of the estimation and analysis of spectra in the frequency domain are the subject of this chapter. Plots of harmonic amplitude or phase versus frequency often result in a more comprehensible presentation of the data or waveform, particularly when the latter is of a random nature. By selecting certain harmonics on some suitable criterion and rejecting others, significant data reduction may be possible. Spectrum analysis has found use in the study of communication engineering signals, event-related or stimulated responses of the human electroencephalogram (EEG) in the diagnosis of brain diseases, other biological signals, meteorological industrial process control and the measurement of noise spectra for optimal linear filters.

The important point to remember here is that the goal of spectral estimation is to describe the distribution (over frequency) of the power contained in a signal, based on a finite set of data. Estimation of power spectra is useful in a variety of applications, including the detection of signals buried in wide-band noise and other applications mentioned above.

The power spectrum of a stationary random process \( x_n \) is mathematically related to the correlation sequence by the discrete-time Fourier transform. In terms of normalized frequency, this is given by

\[
S_{xx}(\omega) = \sum_{m=\infty}^{\infty} R_{xx}(m)e^{-j\omega m}
\]

(3-1)

This can be written as a function of physical frequency \( f \) (e.g., in hertz) by using the relation \( \omega = 2\pi f f_s \), where \( f_s \) is the sampling frequency.
The correlation sequence can be derived from the power spectrum by use of the inverse discrete-time Fourier transform:

\[ S_{xx}(f) = \sum_{m=-\infty}^{\infty} R_{xx}(m)e^{-2\pi jmf_s} \]  

The correlation sequence can be derived from the power spectrum by use of the inverse discrete-time Fourier transform:

\[ R_{xx}(m) = \int_{-\pi}^{\pi} \frac{S_{xx}(\omega)}{2\pi} e^{j\omega m} d\omega = \int_{-f_s/2}^{f_s/2} \frac{S_{xx}(f)e^{2\pi jfm/f_s}}{f_s} df \]  

The average power of the sequence \( x_n \) over the entire Nyquist interval is represented by

\[ R_{xx}(0) = \int_{-\pi}^{\pi} \frac{S_{xx}(\omega)}{2\pi} d\omega = \int_{-f_s/2}^{f_s/2} \frac{S_{xx}(f)}{f_s} df \]  

The quantities

\[ P_{xx}(\omega) = \frac{S_{xx}(\omega)}{2\pi} \quad \text{and} \quad P_{xx}(f) = \frac{S_{xx}(f)}{f_s} \]  

from the above expression are defined as the power spectral density (PSD) of the stationary random signal \( x_n \).

You can see from the above expression that \( P_{xx}(\omega) \) represents the power content of a signal in an infinitesimal frequency band, which is why we call it the power spectral density.

The units of the PSD are power (e.g., watts) per unit of frequency. In the case of \( P_{xx}(\omega) \), this is watts/rad/sample or simply watts/rad. In the case of \( P_{xx}(f) \), the units are watts/hertz. Integration of the PSD with respect to frequency yields units of watts, as expected for the average power \( \bar{P}_{[\omega_1, \omega_2]} \).

For real signals, the PSD is symmetric about DC, and thus \( P_{xx}(\omega) \) for \( 0 \leq \omega < \pi \) is sufficient to completely characterize the PSD.
The spectrum estimation techniques available can be categorized as non-parametric or parametric (MatLab includes a third one called subspace). The non-parametric methods include the periodogram, the Welch modified periodogram, and the Blackman-Tukey methods. All these methods have the advantage of possible implementation using the FFT, but with the disadvantage in the case of short data lengths of limited frequency resolution. Also, considerable care has to be exercised to obtain meaningful result. Parametric methods on the other hand can provide high resolution in addition to being computationally efficient. However, it is necessary to form a sufficiently accurate model of the process from which to estimate the spectrum. The most common parametric approach is to derive the spectrum from the parameters of an autoregressive model of the signal.

A number of pitfalls have to be avoided in performing non-parametric spectral analysis and the associated topics of aliasing, scalloping loss, finite data length, spectral leakage, are discussed in Section 3.3. The deleterious effects of spectral leakage and smearing can be minimized by windowing the data using a suitable window function.

Judgments of the quality of spectral estimates are based on estimation theory. Statistic estimation involves determining the expected values of statistical quantities derived from samples of the population.

### 3.2 Principles of spectrum estimation

In this section a voltage waveform, plotted against time, will be considered initially. The shape of the waveform itself may provide useful information. For example, it may be a sine wave which can obviously be characterized by its amplitude, frequency and phase angle. As an alternative to representing the waveform as a plot of voltage versus time, we could plot the amplitude versus frequency and the phase versus frequency. It can be shown by Fourier analysis that all waveforms can be represented mathematically as the summation of a number of sinusoidal waveforms each with a specific amplitude and phase at its specific frequency. These plots are known as the amplitude and phase spectra. These spectra are important because they provide a complementary way of representing the waveform which more clearly reveals
information about the frequency content of the waveform. The observed shapes of the spectra and changes in them are often helpful in the understanding and interpretation of the waveforms. Amplitude and phase spectra often provide more useful information waveforms.

The transformation of periodic waveforms to the frequency domain by the Fourier series was discussed in an earlier chapter. Non-periodic but continuous signals can be transformed from the time to the frequency domain using the Fourier transform. The amplitude of this transform was shown to have the dimension of V-Hz\(^{-1}\) and when plotted against frequency represents the amplitude spectral density. Thus the area under the curve between two frequencies yields the average voltage of the waveform for those frequency components lying between the two frequencies. Squaring the calculated Fourier transform amplitudes gives the energy spectral density of the voltage waveform in J-Hz\(^{-1}\). The term 'spectrum' is often used to refer to plots of the energy spectral density versus frequency.

If the waveform under investigation is long compared with the time interval over which it may be regarded as having constant statistical moments, (i.e. mean) then the estimate is likely to be inaccurate. This will also be the case when there is a large noise component in the waveform. It is then desirable to smooth the estimated spectrum to obtain an improved estimate. The purpose of spectrum smoothing is essential to remove randomness. The signal-to-noise ratio in random waveforms can be improved by averaging the waveforms, the signal-to-noise ratio being improved by the sqrt K where K waveforms are averaged. Thus one method of improving the accuracy of the estimated spectrum consists of dividing the data into K equal length sections, determining the spectrum of each section, and then averaging the spectra. In this way the average amplitude and average phase of each harmonic frequency component of the K spectra are obtained and plotted to yield the average amplitude and phase spectra. The accuracy of the spectra may be obtained in terms of their variance. For example, the smaller the variance of the power spectral density, the more accurate its estimate. It is therefore important to know what is the effect of a spectrum estimation methods on the variance of the spectrum. Spectrum estimation by this averaging process is called the modified periodogram method. However, sectioning the data results in fewer data per FFT and consequently a coarser spectrum. This disadvantage may be overcome by augmenting zeros. It is therefore always necessary to bear in
mind the opposing claims of the accuracy of the estimate and the required spectral smoothness, and to design for the best compromise. Another approach to smoothing the periodogram is to calculate it from the DFT of the windowed auto-correlation function of the data, and this is known as the Blackman-Tukey method. Since the autocorrelation function of the data consists of the average of the sums of products of the data by themselves at different lags, the SNR is improved.

Data windows also exert a smoothing effect on spectra. In particular windows with small side lobes in the frequency domain filter out noise which falls outside the main lobe and will offer improved smoothing. In fact this type of spectral smoothing is sometimes performed by convolving the spectrum of the data with that of the chosen window function.

Parametric and more recent methods of spectrum estimation are less of an art than the non-parametric methods and may be automated. The concepts of forming models of the data in terms of model parameters and of obtaining the spectra from the frequency response functions of linear systems in terms of these models are outlined as are more recent methods.

3.3 Traditional Methods

In Section 3.3.1 various pitfalls found in spectral analysis are detailed and explanations of how they may be overcome are provided. The technique of windowing and the properties of data windows are described in Section 3.3.1.2. Some properties of spectra and spectral smoothing are described in Section 3.3.1.4.

3.3.1 Pitfalls

3.3.1.1 Sampling rate and aliasing
Before the spectral analysis can be carried out, the first procedure must be to pass the analog signals through an anti-aliasing filter, the function of which is to prevent aliasing of the sampled signal after the following stage of analog-to-digital conversion.
3.3.1.2 Scallop loss or picket-fence effect

As explained previously, the DFT consists of harmonic amplitude and phase components regularly spaced in frequency. The spacing of the spectral lines decreases with the length of the sampled waveform. If, therefore, there is a signal component which falls between two adjacent harmonic frequency components in the spectrum then it cannot be properly represented. Its energy will be shared between neighboring harmonics and the nearby spectral 'amplitudes' will be distorted. A solution to this difficulty lies in arranging for the harmonic components to be more closely spaced and coincident with the signal frequencies. This may be achieved by adding additional data in the form of zeros to the true data. The added zeros are termed augmenting zeros and serve to increase the fidelity of the estimated spectrum to the true spectrum without adding additional information.

As already stated, the effect of finite data length is to limit the achievable frequency resolution. This results in a coarse spectrum which may be rendered smooth and continuous by supplementing the data with augmenting zeros. The process is simply one of interpolation of the spectral curve between adjacent harmonics. A real improvement in resolution can only be achieved if a longer realization is available.

3.3.1.3 Trend removal

Any trends (e.g. linear) in the data must be removed prior to computation of the spectrum because error terms owing to addition of the trend to the data will be integrated and produce large errors in the estimated spectrum.

3.3.1.4 Spectral leakage and spectral smearing

The FFT of a set of sampled data is not the true FFT of the process from which data was obtained. This is because the process is continuous whereas the data is the sampled values of a realization which is truncated at its beginning and end. Without further processing the spectral result is that we appear to have multiplied our spectra with a rectangular window which
introduces spurious peaks into the computed spectrum owing to the effect of the side lobes. This will be true of each frequency component of the signal, and so the amplitude spectrum of the signal will be distorted by the addition and subtraction of the large number of window side and main lobes. The effect may be to introduce spurious peaks, or to conceal true peaks in the spectrum, the phenomenon being known as spectral leakage. To minimize this it is necessary to modify the data by multiplying them by a window shaped to reduce the side lobe effect. Suitable windows have a value of 1 at the mid-data point and tapered to zero at \( n = 0 \) and \( n = N - 1 \). We have discussed some of the available windows previously.

In order to minimize spectral leakage the window shape is chosen to minimize its side lobe levels. Unfortunately, this has the effect of increasing the main lobe width, causing it to spread into the adjacent side lobes. This therefore aliases the side lobes, and is repeated at each harmonic frequency, the overall result being an aliased signal spectrum, known as smearing. Windows and their parameters therefore have to be carefully chosen in order to strike the optimum balance between resolution and the statistical accuracy of the spectral estimate. Studies of windows have shown that preferred windows are Blackman-Harris, Dolph-Chebshev and Kaiser-Bessel.

### 3.4 The periodogram method and periodogram properties

One way of estimating the power spectrum of a process is to simply find the discrete-time Fourier transform of the samples of the process (usually done on a grid with an FFT) and take the magnitude squared of the result. This estimate is called the *periodogram*.

The periodogram estimate of the PSD of a length-\( L \) signal \( x_L[n] \) is

\[
P_{xx}(f) = \frac{|X_L(f)|^2}{f_s L}
\]

where

\[
X_L(f) = \sum_{n=-L}^{L-1} x_L[n] e^{-2\pi j fn / f_s}
\]

(3-6)

(3-7)
The actual computation of $X_L(f)$ can be performed only at a finite number of frequency points, $N$, and usually employs the FFT. In practice, most implementations of the periodogram method compute the $N$-point PSD estimate

$$
\hat{P}_{xx}[f_k] = \frac{|X_L[f_k]|^2}{f_s L}, \quad f_k = \frac{k f_s}{N}, \quad k = 0, 1, \ldots, N - 1
$$

(3-8)

where

$$
X_L[f_k] = \sum_{n=0}^{N-1} x_L[n] e^{-\pi j k n / N}
$$

(3-9)

It is wise to choose $N > L$ so that $N$ is the next power of two larger than $L$. To evaluate $X_L[f_k]$, we simply pad $x_L[n]$ with zeros to length $N$. If $L > N$, we must wrap $x_L[n]$ modulo-$N$ prior to computing $X_L[f_k]$.

As an example, consider the following 1001-element signal $x_n$, which consists of two sinusoids plus noise:

```matlab
randn('state',0);
fs = 1000; % Sampling frequency
t = (0:fs)/fs; % One second worth of samples
A = [1 2]; % Sinusoid amplitudes (row vector)
f = [150;140]; % Sinusoid frequencies (column vector)
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
```

Note the three last lines above illustrate a convenient and general way to express the sum of sinusoids. Together they are equivalent to

$$
x_n = \sin(2\pi f_1 t) + 2\sin(2\pi f_2 t) + 0.1\text{randn(size(t))};
$$
The periodogram estimate of the PSD can be computed by

\[
P_{xx} = \text{periodogram}(x_n, [], 'twosided', 1024, fs); \quad \% \text{note the second argument is the window function}
\]

PERIODOGRAM(X, WINDOW, 'twosided', NFFT, Fs) returns a two-sided PSD of a real signal X. WINDOW must be a vector of the same length as X. If WINDOW is a window other than a boxcar (rectangular), the resulting estimate is a modified periodogram. If WINDOW is specified as empty, the default window (boxcar) is used. In this case, Pxx will have length NFFT and will be computed over the interval \([0, 2\pi)\) if Fs (sampling frequency) is not specified and over the interval \([0, Fs)\) if Fs is specified. Alternatively, the string 'twosided' can be replaced with the string 'onesided' for a real signal X. The string 'twosided' or 'onesided' may be placed in any position in the input argument list after WINDOW. NFFT specifies the number of FFT points used to calculate the PSD estimate. For real X, Pxx has length \((NFFT/2+1)\) if NFFT is even, and \((NFFT+1)/2\) if NFFT is odd. For complex X, Pxx always has length NFFT. If NFFT is specified as empty, the default NFFT is used (FFT of length given by the larger of 256 and the next power of 2 greater than the length of X).

and a plot of the estimate can be displayed by simply omitting the output argument, as below:

\[
\text{periodogram}(x_n, [], 'twosided', 1024, fs);
\]

![Fig. (3.1) Periodogram PSD Estimate](image-url)
The average power can be computed by approximating the integral with the following sum:

$$\text{Pow} = \left( \frac{f_s}{\text{length}(Pxx)} \right) \times \text{sum}(Pxx)$$

$$\text{Pow} = 2.4989$$

We can also compute the average power from the one-sided PSD estimate:

$$Pxxo = \text{periodogram}(x_n, [], 1024, f_s);$$

$$\text{Pow} = \left( \frac{f_s}{2 \times \text{length}(Pxxo)} \right) \times \text{sum}(Pxxo)$$

$$\text{Pow} = 2.4979$$

Performance of the Periodogram. The following sections discuss the performance of the periodogram with regard to the issues of leakage, resolution, bias, and variance.

**Spectral Leakage.** Consider the power spectrum or PSD of a finite-length signal $x_L[n]$, as discussed in the Periodogram. It is frequently useful to interpret $x_L[n]$ as the result of multiplying an infinite signal, $x[n]$, by a finite-length rectangular window, $w_R[n]$:

$$x_L[n] = x[n] \cdot w_R[n] \quad (3-10)$$

Because multiplication in the time domain corresponds to convolution in the frequency domain, the Fourier transform of the expression above is

$$X_L(f) = \frac{1}{f_s} \int_{-f_s/2}^{f_s/2} X(\rho) W_R(f - \rho) d\rho \quad (3-11)$$

The expression developed earlier for the periodogram,

$$\hat{P}_{xx}(f) = \frac{|X_L(f)|^2}{f_sL} \quad (3-12)$$

illustrates that the periodogram is also influenced by this convolution.
The effect of the convolution is best understood for sinusoidal data. Suppose that $x[n]$ is composed of a sum of $M$ complex sinusoids:

$$x[n] = \sum_{k=1}^{M} A_k e^{j\omega_k n}$$  \hspace{2cm} (3-13)

Its spectrum is

$$X(f) = f_s \sum_{k=1}^{M} A_k \delta(f - f_k)$$  \hspace{2cm} (3-14)

which for a finite-length sequence becomes

$$X_L(f) = \frac{f_s}{2} \left( \sum_{k=1}^{M} A_k \delta(f - f_k) \right) W_R(f - \rho) d\rho = \sum_{k=1}^{M} A_k W_R(f - f_k)$$  \hspace{2cm} (3-15)

So in the spectrum of the finite-length signal, the Dirac deltas have been replaced by terms of the form $W_R(f - f_k)$, which corresponds to the frequency response of a rectangular window centered on the frequency $f_k$.

The frequency response of a rectangular window has the shape of a sinc signal, as shown below.

```matlab
xn = [ones(50,1)', zeros(50,1)'];
plot(xn);
```

![Fig.(3.2) Plot of sequence x[n]](image-url)
periodogram(xn, [], 'twosided', 1024, 1000);

The plot displays a main lobe and several side lobes, the largest of which is approximately 13.5 dB below the main lobe peak. These lobes account for the effect known as spectral leakage. While the infinite-length signal has its power concentrated exactly at the discrete frequency points $f_k$, the windowed (or truncated) signal has a continuum of power "leaked" around the discrete frequency points $f_k$.

Because the frequency response of a short rectangular window is a much poorer approximation to the Dirac delta function than that of a longer window, spectral leakage is especially evident when data records are short. Consider the following sequence of 100 samples:

```matlab
fs = 1000;           % Sampling frequency
fs = (0:fs/10)/fs;   % One-tenth of a second worth of samples
A = [1 2];           % Sinusoid amplitudes
f = [150;140];       % Sinusoid frequencies
xn = A*sin(2*pi*f*t); 
plot(xn); 
```

Fig. (3.3) Periodogram of sequence x[n]
Note that where we expect two frequency spikes at 140 and 150 Hz and nothing else, we see side lobes where we have leakage due to the finite length of the data.

It is important to note that the effect of spectral leakage is contingent solely on the length of the data record. It is not a consequence of the fact that the periodogram is computed at a finite number of frequency samples.
Resolution. Resolution refers to the ability to discriminate spectral features, and is a key concept in the analysis of spectral estimator performance.

In order to resolve two sinusoids that are relatively close together in frequency, it is necessary for the difference between the two frequencies to be greater than the width of the mainlobe of the leaked spectra for either one of these sinusoids. The mainlobe width is defined to be the width of the mainlobe at the point where the power is half the peak mainlobe power (i.e., 3 dB width). This width is approximately equal to $\frac{f}{L}$. In other words, for two sinusoids of frequencies $f_1$ and $f_2$, the resolvability condition requires that

$$\Delta f = (f_1 - f_2) > \frac{f_s}{L}$$

(3-16)

In the example above, where two sinusoids are separated by only 10 Hz, the data record must be greater than 100 samples to allow resolution of two distinct sinusoids by a periodogram. Consider a case where this criterion is not met, as for the sequence of 67 samples below:

```matlab
randn('state',0) % We will add some random noise in also
fs = 1000; % Sampling frequency
t = (0:fs/15)./fs; % 67 samples
A = [1 2]; % Sinusoid amplitudes
f = [150;140]; % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
plot(xn);
```
Fig. (3.6) Plot of sequence x[n]

periodogram(xn,[],1024,fs);

Fig. (3.7) Periodogram of sequence x[n]

Note here we have lost the separate frequency peaks for the 140 and 150 Hz sinusoids as they are lumped together into a broad frequency peak.
The above discussion about resolution did not consider the effects of noise since the signal-to-noise ratio (SNR) has been relatively high thus far. When the SNR is low, true spectral features are much harder to distinguish, and noise artifacts appear in spectral estimates based on the periodogram. The example below illustrates this:

```matlab
randn('state',0)
fs = 1000;
t = (0:fs/10)/fs; % Back to 100 samples
A = [1 2];
f = [150;140];
xn = A*sin(2*pi*f*t) + 2*randn(size(t)); % Larger amplitude of noise
plot(xn);
```

Fig.(3.8) Plot of sequence x[n]

```matlab
periodogram(xn,[],1024,fs);
```
Note here we have again lost the separate frequency peaks for the 140 and 150 Hz sinusoids as they are lumped together into a broad frequency peak, but this time due to the magnitude of the noise present.

The Modified Periodogram  The modified periodogram windows the time-domain signal prior to computing the FFT in order to smooth the edges of the signal. This has the effect of reducing the height of the sidelobes or spectral leakage. This phenomenon gives rise to the interpretation of sidelobes as spurious frequencies introduced into the signal by the abrupt truncation that occurs when a rectangular window is used. For nonrectangular windows, the end points of the truncated signal are attenuated smoothly, and hence the spurious frequencies introduced are much less severe. On the other hand, nonrectangular windows also broaden the mainlobe, which results in a net reduction of resolution.

The periodogram function allows you to compute a modified periodogram by specifying the window to be used on the data. For example, compare a rectangular window and a Hamming window:
randn('state',0)
fs = 1000; % Sampling frequency
t = (0:fs/10)/fs; % One-tenth of a second worth of samples
A = [1 2]; % Sinusoid amplitudes
f = [150;140]; % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
periodogram(xn,rectwin(length(xn)),1024,fs);
We can verify that although the sidelobes are much less evident in the Hamming-windowed periodogram, the two main peaks are wider. In fact, the 3 dB width of the mainlobe corresponding to a Hamming window is approximately twice that of a rectangular window. Hence, for a fixed data length, the PSD resolution attainable with a Hamming window is approximately half that attainable with a rectangular window. The competing interests of mainlobe width and sidelobe height can be resolved to some extent by using variable windows such as the Kaiser window.

Nonrectangular windowing affects the average power of a signal because some of the time samples are attenuated when multiplied by the window. To compensate for this, the periodogram function normalizes the window to have an average power of unity. This way the choice of window does not affect the average power of the signal.

The modified periodogram estimate of the PSD is

$$\hat{P}_{xx}(f) = \frac{|X_L(f)|^2}{f_sLU}$$

(3-17)

where $U$ is the window normalization constant

$$U = \frac{1}{L} \sum_{n=0}^{L-1} |w(n)|^2$$

(3-18)

which is independent of the choice of window. The addition of $U$ as a normalization constant ensures that the modified periodogram is asymptotically unbiased.

**Welch's Method** An improved estimator of the PSD is the one proposed by Welch [8]. The method consists of dividing the time series data into (possibly overlapping) segments, computing a modified periodogram of each segment, and then averaging the PSD estimates. The result is Welch's PSD estimate.

Welch's method is implemented in the Signal Processing Toolbox by the `pwelch` function. By default, the data is divided into eight segments with 50% overlap between them. A Hamming window is used to compute the modified periodogram of each segment. The averaging of
modified periodograms tends to decrease the variance of the estimate relative to a single periodogram estimate of the entire data record. Although overlap between segments tends to introduce redundant information, this effect is diminished by the use of a nonrectangular window, which reduces the importance or weight given to the end samples of segments (the samples that overlap).

However, as mentioned above, the combined use of short data records and nonrectangular windows results in reduced resolution of the estimator. In summary, there is a tradeoff between variance reduction and resolution. One can manipulate the parameters in Welch’s method to obtain improved estimates relative to the periodogram, especially when the SNR is low. This is illustrated in the following example. Consider an original signal consisting of 301 samples:

```matlab
randn('state',1)
fs = 1000;               % Sampling frequency
t = (0:0.3*fs)/fs;       % 301 samples
A = [2 8];               % Sinusoid amplitudes (row vector)
f = [150;140];           % Sinusoid frequencies (column vector)
xn = A*sin(2*pi*f*t) + 5*randn(size(t));
periodogram(xn,rectwin(length(xn)),1024,fs);
```

**Fig. (3.12) Periodogram PSD Estimate**
We can obtain Welch's spectral estimate for 3 segments with 50% overlap with

\[
p\text{welch}(x_n, \text{rectwin}(150), 75, 512, fs);
\]

In the periodogram above, noise and the leakage make one of the sinusoids essentially indistinguishable from the artificial peaks. In contrast, although the PSD produced by Welch's method has wider peaks, you can still distinguish the two sinusoids, which stand out from the "noise floor." However, if we try to reduce the variance further, the loss of resolution causes one of the sinusoids to be lost altogether:

\[
p\text{welch}(x_n, \text{hamming}(100), 75, 512, fs);
\]
Multitaper Method. The periodogram can be interpreted as filtering a length $L$ signal, $x_L[n]$, through a filter bank (a set of filters in parallel) of $L$ FIR bandpass filters. The 3 dB bandwidth of each of these bandpass filters can be shown to be approximately equal to $f_s / L$. The magnitude response of each one of these bandpass filters resembles that of the rectangular window discussed in Spectral Leakage. The periodogram can thus be viewed as a computation of the power of each filtered signal (i.e., the output of each bandpass filter) that uses just one sample of each filtered signal and assumes that the PSD of $x_L[n]$ is constant over the bandwidth of each bandpass filter.

As the length of the signal increases, the bandwidth of each bandpass filter decreases, making it a more selective filter, and improving the approximation of constant PSD over the bandwidth of the filter. This provides another interpretation of why the PSD estimate of the periodogram improves as the length of the signal increases. However, there are two factors apparent from this standpoint that compromise the accuracy of the periodogram estimate. First, the rectangular window yields a poor bandpass filter. Second, the computation of the power at the output of each bandpass filter relies on a single sample of the output signal, producing a very crude approximation.
Welch's method can be given a similar interpretation in terms of a filter bank. In Welch's implementation, several samples are used to compute the output power, resulting in reduced variance of the estimate. On the other hand, the bandwidth of each bandpass filter is larger than that corresponding to the periodogram method, which results in a loss of resolution. The filter bank model thus provides a new interpretation of the compromise between variance and resolution.

Thompson's multitaper method (MTM) builds on these results to provide an improved PSD estimate. Instead of using bandpass filters that are essentially rectangular windows (as in the periodogram method), the MTM method uses a bank of optimal bandpass filters to compute the estimate. These optimal FIR filters are derived from a set of sequences known as discrete prolate spheroidal sequences (DPSSs, also known as Slepian sequences).

In addition, the MTM method provides a time-bandwidth parameter with which to balance the variance and resolution. This parameter is given by the time-bandwidth product, $NW$, and it is directly related to the number of tapers used to compute the spectrum. There are always $2*NW-1$ tapers used to form the estimate. This means that, as $NW$ increases, there are more estimates of the power spectrum, and the variance of the estimate decreases. However, the bandwidth of each taper is also proportional to $NW$, so as $NW$ increases, each estimate exhibits more spectral leakage (i.e., wider peaks) and the overall spectral estimate is more biased. For each data set, there is usually a value for $NW$ that allows an optimal trade-off between bias and variance.

The Signal Processing Toolbox function that implements the MTM method is called \texttt{pmtm}. Use \texttt{pmtm} to compute the PSD of $xn$ from the previous examples:

```matlab
randn('state',0)
fs = 1000;           % Sampling frequency
t = (0:fs)/fs;       % One second worth of samples
A = [1 2];           % Sinusoid amplitudes
f = [150;140];       % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
```
\[ [P,F] = \text{pmtm}(x_n,4,1024,fs); \]
\[ \text{plot}(F,10\cdot\log10(P)) \quad \% \text{Plot in dB/Hz} \]
\[ \text{xlabel('Frequency (Hz)')} \]
\[ \text{ylabel('Power Spectral Density (dB/Hz)')} \]

Fig. (3.15) PSD Estimate

Here we also see a more accurate detection of the frequencies (140 and 150 Hz).

By lowering the time-bandwidth product, you can increase the resolution at the expense of larger variance:

\[ [P1,f] = \text{pmtm}(x_n,3/2,1024,fs); \]
\[ \text{plot}(f,10\cdot\log10(P1)) \]
\[ \text{xlabel('Frequency (Hz)')} \]
\[ \text{ylabel('Power Spectral Density (dB/Hz)')} \]
3.5 Modern parametric estimation methods

The nonparametric methods described in the previous sections of this chapter which utilize periodograms and FFTs are subject to the aforementioned limitations of low spectral resolution in the case of short records and the requirement for windowing to reduce the spectral leakage. These difficulties may be overcome by parametric methods. The price to be paid is an extensive investigation of an appropriate model for each process, a determination of the necessary order of the chosen model for adequate representation of the data, and computation of the model parameters. The advantages gained are increased spectral resolution, applicability to short data lengths, and avoidance of spectral leakage, scalloping loss, spectral smearing, and window biasing effects. Because of the importance of these parametric techniques, the most commonly used method of autoregressive modeling is presented. Nevertheless, although an improvement over the nonparametric methods described, these parametric methods do have some disadvantages which may be avoided by alternative modern approaches such as the sequential or adaptive and the maximum likelihood methods. To summarize, the parametric approach calls for parametric modeling of the data, a well-established branch of time series analysis, combined with an interpretation of the data as being the output of a linear system excited by white noise. This system is represented by a polynomial transfer function expressed in terms of the model parameters. The spectrum of the data is computed from this transfer function.
**Parametric PSD Methods.** Parametric methods can yield higher resolutions than nonparametric methods in cases when the signal length is short. These methods use a different approach to spectral estimation; instead of trying to estimate the PSD directly from the data, they *model* the data as the output of a linear system driven by white noise, and then attempt to estimate the parameters of that linear system.

The most commonly used linear system model is the *all-pole model*, a filter with all of its zeroes at the origin in the \(z\)-plane. The output of such a filter for white noise input is an autoregressive (AR) process. For this reason, these methods are sometimes referred to as *AR methods* of spectral estimation.

The AR methods tend to adequately describe spectra of data that is "peaky," that is, data whose PSD is large at certain frequencies. The data in many practical applications (such as speech) tends to have "peaky spectra" so that AR models are often useful. In addition, the AR models lead to a system of linear equations which is relatively simple to solve.

The Signal Processing Toolbox offers the following AR methods for spectral estimation:

- **Yule-Walker AR method (autocorrelation method)**
- **Burg method**
- **Covariance method**
- **Modified covariance method**

All AR methods yield a PSD estimate given by

\[
\hat{P}_{AR}(f) = \frac{1}{f_s} \frac{\varepsilon_p}{1 + \sum_{k=1}^{p} \hat{a}_p(k)e^{-2\pi kf/f_s}}^2
\]

(3-19)

The different AR methods estimate the AR parameters \(a_p(k)\) slightly differently, yielding different PSD estimates. The following table provides a summary of the different AR methods.
<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Burg</th>
<th>Covariance</th>
<th>Modified Covariance</th>
<th>Yule-Walker</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Does not apply window to data</td>
<td>Does not apply window to data</td>
<td>Does not apply window to data</td>
<td>Applies window to data</td>
</tr>
<tr>
<td></td>
<td>Minimizes the forward and backward</td>
<td>Minimizes the forward prediction error in the</td>
<td>Minimizes the forward and backward prediction</td>
<td>Minimizes the forward prediction error in the</td>
</tr>
<tr>
<td></td>
<td>prediction errors in the least squares</td>
<td>least squares sense</td>
<td>prediction errors in the least squares</td>
<td>least squares sense</td>
</tr>
<tr>
<td></td>
<td>sense, with the AR coefficients constrained</td>
<td></td>
<td>sense</td>
<td>(also called &quot;Autocorrelation method&quot;)</td>
</tr>
<tr>
<td></td>
<td>to satisfy the L-D recursion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Advantages</td>
<td>High resolution for short data records</td>
<td>Better resolution than Y-W for short data</td>
<td>High resolution for short data records</td>
<td>Performs as well as other methods for large</td>
</tr>
<tr>
<td></td>
<td></td>
<td>records (more accurate estimates)</td>
<td></td>
<td>data records</td>
</tr>
<tr>
<td></td>
<td>Always produces a stable model</td>
<td>Able to extract frequencies from data</td>
<td>Able to extract frequencies from data</td>
<td>Always produces a stable model</td>
</tr>
<tr>
<td></td>
<td></td>
<td>consisting of p or more pure sinusoids</td>
<td>consisting of p or more pure sinusoids</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Does not suffer spectral line-splitting</td>
</tr>
<tr>
<td>Disadvantages</td>
<td>Peak locations highly dependent on initial</td>
<td>May produce unstable models</td>
<td>May produce unstable models</td>
<td>Performs relatively poorly for short data</td>
</tr>
<tr>
<td></td>
<td>phase</td>
<td></td>
<td></td>
<td>records</td>
</tr>
<tr>
<td></td>
<td>May suffer spectral line-splitting for</td>
<td>Frequency bias for estimates of sinusoids</td>
<td>Peak locations slightly dependent on initial</td>
<td>Frequency bias for estimates of sinusoids</td>
</tr>
<tr>
<td></td>
<td>sinusoids in noise, or when order is very</td>
<td>in noise</td>
<td>phase</td>
<td>in noise</td>
</tr>
<tr>
<td></td>
<td>large</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Frequency bias for estimates of</td>
<td>Minor frequency bias for estimates of</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sinusoids in noise</td>
<td>sinusoids in noise</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conditions for</td>
<td>Order must be less than or equal to half</td>
<td>Order must be less than or equal to 2/3 the</td>
<td>Because of the biased estimate, the</td>
<td></td>
</tr>
<tr>
<td>Nonsingularity</td>
<td>the input frame size</td>
<td>input frame size</td>
<td>autocorrelation matrix is guaranteed to</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>positive-definite, hence nonsingular</td>
<td></td>
</tr>
</tbody>
</table>
Yule-Walker AR Method. The Yule-Walker AR method of spectral estimation computes the AR parameters by forming a biased estimate of the signal's autocorrelation function, and solving the least squares minimization of the forward prediction error. This results in the Yule-Walker equations.

\[
\begin{bmatrix}
    r(1) & r(2)^* & \cdots & r(p)^* \\
    r(2) & r(1) & \cdots & r(p-1)^* \\
    \vdots & \ddots & \ddots & \vdots \\
    r(p) & \cdots & r(2) & r(1)
\end{bmatrix}
\begin{bmatrix}
    \alpha(2) \\
    \alpha(3) \\
    \vdots \\
    \alpha(p+1)
\end{bmatrix}
= \begin{bmatrix}
    -r(2) \\
    -r(3) \\
    \vdots \\
    -r(p+1)
\end{bmatrix}
\]

(3-20)

The use of a biased estimate of the autocorrelation function ensures that the autocorrelation matrix above is positive definite. Hence, the matrix is invertible and a solution is guaranteed to exist. Moreover, the AR parameters thus computed always result in a stable all-pole model. The Yule-Walker equations can be solved efficiently via Levinson's algorithm, which takes advantage of the Toeplitz structure of the autocorrelation matrix. The toolbox function `pyulear` implements the Yule-Walker AR method. For example, compare the spectrum of a speech signal using Welch's method and the Yule-Walker AR method:

```matlab
load mtlb;  % speech signal data
plot(mtlb(1:512));
```

![Fig.(3.17) Speech signal data](image)
\[ [P1,f] = \text{pwelch}(\text{mtlb}, \text{hamming}(256), 128, 1024, \text{fs}); \]
\[ [P2,f] = \text{pyulear}(\text{mtlb}, 14, 1024, \text{fs}); \]
\[ \text{plot}(f, 10 \times \log_{10}(P1), ':', f, 10 \times \log_{10}(P2)); \text{ grid} \]
\[ \text{ylabel('PSD Estimates (dB/Hz)');} \]
\[ \text{xlabel('Frequency (Hz)');} \]
\[ \text{legend('Welch','Yule-Walker AR')}; \]

The solid-line Yule-Walker AR spectrum is smoother than the periodogram because of the simple underlying all-pole model.

**Burg Method.** The Burg method for AR spectral estimation is based on minimizing the forward and backward prediction errors while satisfying the Levinson-Durbin recursion. In contrast to other AR estimation methods, the Burg method avoids calculating the autocorrelation function, and instead estimates the reflection coefficients directly.

The primary advantages of the Burg method are resolving closely spaced sinusoids in signals with low noise levels, and estimating short data records, in which case the AR power spectral density estimates are very close to the true values. In addition, the Burg method ensures a stable AR model and is computationally efficient.
The accuracy of the Burg method is lower for high-order models, long data records, and high signal-to-noise ratios (which can cause line splitting, or the generation of extraneous peaks in the spectrum estimate). The spectral density estimate computed by the Burg method is also susceptible to frequency shifts (relative to the true frequency) resulting from the initial phase of noisy sinusoidal signals. This effect is magnified when analyzing short data sequences. The toolbox function `pburg` implements the Burg method. Compare the spectrum of the speech signal generated by both the Burg method and the Yule-Walker AR method. They are very similar for large signal lengths:

```matlab
load mtlb
[P1,f] = pburg(mtlb(1:512),14,1024,fs); % 14th order model
[P2,f] = pyulear(mtlb(1:512),14,1024,fs); % 14th order model
plot(f,10*log10(P1),':',f,10*log10(P2)); grid
ylabel('Magnitude (dB)'); xlabel('Frequency (Hz)');
legend('Burg','Yule-Walker AR')
```

![Fig. (3.19) Burg and Yule-Walker AR methods](image)

Compare the spectrum of a noisy signal computed using the Burg method and the Welch method:

```matlab
randn('state',0)
fs = 1000; % Sampling frequency
t = (0:fs)/fs; % One second worth of samples
```
A = [1 2]; % Sinusoid amplitudes
f = [150;140]; % Sinusoid frequencies
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
[P1,f] = pwelch(xn,hamming(256),128,1024,fs);
[P2,f] = pburg(xn,14,1024,fs);
plot(f,10*log10(P1),':',f,10*log10(P2)); grid
ylabel('PSD Estimates (dB/Hz)');
xlabel('Frequency (Hz)');
legend('Welch','Burg')

Note that, as the model order for the Burg method is reduced, a frequency shift due to the initial phase of the sinusoids will become apparent.

Covariance and Modified Covariance Methods. The covariance method for AR spectral estimation is based on minimizing the forward prediction error. The modified covariance method is based on minimizing the forward and backward prediction errors. The toolbox functions pcov and pmcov implement the respective methods. Compare the spectrum of the speech signal generated by both the covariance method and the modified covariance method. They are nearly identical, even for a short signal length:
load mtlb

\[ [P1,f] = \text{pcov}(\text{mtlb}(1:64),14,1024,fs); \] % 14th order model

\[ [P2,f] = \text{pmcov}(\text{mtlb}(1:64),14,1024,fs); \] % 14th order model

\text{plot}(f,10*\log10(P1),':',f,10*\log10(P2)); \text{grid}

\text{ylabel('Magnitude (dB')}}; \text{xlabel('Frequency (Hz')}};

\text{legend('Covariance','Modified Covariance')}

**Fig.(3.21) Covariance and Modified Covariance Methods**

**MUSIC and Eigenvector Analysis Methods.** The \text{pmusic} and \text{peig} functions provide two related spectral analysis methods: \text{pmusic} provides the multiple signal classification (MUSIC) method developed by Schmidt, while \text{peig} provides the eigenvector (EV) method developed by Johnson.

Both of these methods are frequency estimator techniques based on eigenanalysis of the autocorrelation matrix. This type of spectral analysis categorizes the information in a correlation or data matrix, assigning information to either a signal subspace or a noise subspace.

**Eigenanalysis Overview.** Consider a number of complex sinusoids embedded in white noise. You can write the autocorrelation matrix \( R \) for this system as the sum of the signal autocorrelation matrix \( S \) and the noise autocorrelation matrix \( W \):

\[
R = S + W
\]
There is a close relationship between the eigenvectors of the signal autocorrelation matrix and the signal and noise subspaces. The eigenvectors $v$ of $S$ span the same signal subspace as the signal vectors. If the system contains $M$ complex sinusoids and the order of the autocorrelation matrix is $p$, eigenvectors $v_{M+1}$ through $v_{p+1}$ span the noise subspace of the autocorrelation matrix.

**Frequency Estimator Functions.** To generate their frequency estimates, eigenanalysis methods calculate functions of the vectors in the signal and noise subspaces. Both the MUSIC and EV techniques choose a function that goes to infinity (denominator goes to zero) at one of the sinusoidal frequencies in the input signal. Using digital technology, the resulting estimate has sharp peaks at the frequencies of interest; this means that there might not be infinity values in the vectors. The MUSIC estimate is given by the formula

$$
P_{music}(f) = \frac{1}{\sum_{k = p + 1}^{N} v_k v_k^H e(f)} = \frac{1}{\sum_{k = p + 1}^{N} |v_k^H e(f)|^2}
$$

(3-22)

where $N$ is the size of the eigenvectors and $e(f)$ is a vector of complex sinusoids.

$$
e(f) = [1 \, \exp(j2\pi f) \, \exp(j2\pi f \cdot 2) \, \exp(j2\pi f \cdot 4) \, \ldots \, \exp(j2\pi f \cdot (n - 1))]^H
$$

(3-23)

$v$ represents the eigenvectors of the input signal's correlation matrix; $v_k$ is the $k^{th}$ eigenvector. $H$ is the conjugate transpose operator. The eigenvectors used in the sum correspond to the smallest eigenvalues and span the noise subspace ($p$ is the size of the signal subspace).

The expression $v_k^H e(f)$ is equivalent to a Fourier transform (the vector $e(f)$ consists of complex exponentials). This form is useful for numeric computation because the FFT can be computed for each $v_k$ and then the squared magnitudes can be summed.

The EV method weights the summation by the eigenvalues of the correlation matrix:
\[
P_{ev}(f) = \frac{1}{N} \sum_{k=p+1}^{N} \left| v_k^H e(f) \right|^2 / \lambda_k
\]  

(3-24)

The \texttt{pmusic} and \texttt{peig} functions in this toolbox use the \texttt{svd} (singular value decomposition) function in the signal case and the \texttt{eig} function for analyzing the correlation matrix and assigning eigenvectors to the signal or noise subspaces. When \texttt{svd} is used, \texttt{pmusic} and \texttt{peig} never compute the correlation matrix explicitly, but the singular values are the eigenvalues.

\begin{verbatim}
randn('state',0);
fs = 1000;  % Sampling frequency
t = (0:fs)/fs;  % One second worth of samples
A = [1 2];  % Sinusoid amplitudes (row vector);
f = [150;140];  % Sinusoid frequencies (column vector)
xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
periodogram(xn,[],1024,fs);
\end{verbatim}

\texttt{X2=corrmtx(xn,20,'cov');}  % Estimate the correlation matrix using the covariance method.
\texttt{pmusic(X2,4,fs)}  % Use twice the signal space dimension for real sinusoids.

\textbf{Fig.(3.22) Periodogram PSD Estimate}
peig(X2,4,fs);

Fig. (3.23) Music Pseudospectrum

Fig. (3.24) Eigenvector Method Pseudospectrum
3.6 Theoretical Guidelines for Spectrum Estimation

Spectrum estimation of periodic and random signals is one of the important application areas of digital signal processing (DSP). Speech recognition problems use spectrum analysis as a preliminary measurement to perform speech bandwidth reduction and further acoustic processing. Sonar systems use sophisticated spectrum analysis to locate submarines and surface vessels. Spectral measurements in radar are used to obtain target location and velocity information. To this end, spectrum estimation and analysis techniques played a central role in signal processing research over the past several decades.

A spectrum is a relationship typically represented by a plot of the magnitude or relative value of some parameter against frequency. Every physical phenomenon, whether it be an electromagnetic, thermal, mechanical, hydraulic or any other system, has a unique spectrum associated with it. In electronics, the phenomenon are dealt with in terms of signals, represented as fixed or varying electrical quantities of voltage, current and power. These quantities are typically described in the time domain and for every function of time f(t), an equivalent frequency domain function F(ω) can be found that specifically describes the frequency-component content (frequency spectrum) required to generate f(t). A study of relationships between the time domain and its corresponding frequency domain representation is the subject of Fourier analysis and Fourier transforms [1].

The forward Fourier transform, time to frequency domain, of the function x(t) is defined

\[ F[x(t)] = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt = X(\omega) \] (3-25)

and the inverse Fourier transform, frequency to time domain, of X(ω) is

\[ F^{-1}[X(\omega)] = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)e^{j\omega t} d\omega = x(t) \] (3-26)

Though these expressions are in themselves self-explanatory, a short illustrative example will be presented to aid in relating the two domains.
If an arbitrary time function representation of a periodic electrical signal, \( f(t) \), were plotted versus time as shown in Figure 3.25, its Fourier transform would indicate a spectral content consisting of a DC component, a fundamental frequency component \( \omega_0 \), a fifth harmonic component \( 5\omega_0 \) and a ninth harmonic component \( 9\omega_0 \) (see Figure 3.26). It is explanatory in Figure 3.27 that the superposition of these frequency components, in fact, yields the original time function \( f(t) \).

Fig. (3.25) An Electrical Signal \( f(t) \)

Fig. (3.26) Spectral Composition or Spectrum \( F(\omega) \) or \( f(t) \).
3.6.1 Non-Parametric Spectrum Estimation Methods

A simplest way to estimate the power spectrum of a signal is to use non-parametric (window-based) methods which are also known as Classical methods. Window-based methods are the most fundamental type of spectral estimation approaches without requiring signal modeling. One of the most attractive properties of the window-based methods is their simplicity for implementation. However, these methods suffer from low frequency resolution especially when the length of observation data is short.

**Periodograms**

A most fundamental method in the family of window-based spectral estimation methods is the so-called ‘periodogram’. For a given data sequence x(n), n=0,1,…,N-1, spectral estimate from the periodogram can be described as,

$$
\hat{P}_{\text{per}}(e^{j\omega}) = \sum_{k=-N+1}^{N-1} r_k (k) e^{-jk\omega}
$$

(3-27)

Which is directly linked to $X_N(e^{j\omega})$, the DTFT of x(n), as follows,

$$
\hat{P}_{\text{per}}(e^{j\omega}) = \frac{1}{N} \left| X_N(e^{j\omega}) \right|^2
$$

(3-28)

---

Fig. (3.27) Combined Time Domain and Frequency Domain Plots
The periodogram has rather a poor performance, since it is an inconsistent estimate of the true power spectrum (asymptotically unbiased, variance is approximately equal to the power of the signal, i.e., \( \text{Var}(\hat{P}_\text{per}(e^{j\omega})) \approx P^2(e^{j\omega}) \)), and the frequency resolution is limited by the mainlobe width of the rectangular window, i.e., \( \Delta\omega_{3,db} = 0.89 \frac{2\Pi}{N} \). A further disadvantage is that the low attenuation of sidelobe in the rectangular window can lead to so-called ‘masking effect’ where a weaker signal component can be masked by a neighbouring strong frequency component.

The modified periodogram, replacing a rectangular window by another window of choice, may reduce the masking effect however at the price of reduced frequency resolution (which is dependent on the selected window).

**Bartlett’s method (periodogram averaging)**

To overcome the disadvantage of inconsistent estimate in the periodogram or the modified periodogram, Bartlett’s method divides data into \( K \) segments: for each segments of data a periodogram is applied. This is followed by averaging all periodograms as follows:

\[
\hat{P}_\text{B}(e^{j\omega}) = \frac{1}{K} \sum_{i=1}^{K} \hat{P}_{i
\text{per}}(e^{j\omega})
\]

(3-29)

This leads to a consistent estimate of power spectrum (in this occasion, if the number of segments approaches to finite, the variance of the estimate approaches to zero). The price for this is the reduced frequency resolution (due to a shorter data length in each segment), which is

\[
\Delta\omega_{3,db} = 0.89K \frac{2\Pi}{N} .
\]

Welch’s method, using modified periodogram averaging and overlapped data segments, is another alternative.
Blackman-Tukey’s method

Viewing that the variance for the estimated autocorrelation sequence \( \{r_x(k)\} \) increases with the increase of lag \( k \), which is caused by using a smaller number of data samples used for estimating the autocorrelation value when \( k \) becomes large. To make the power spectrum estimate more reliable, an additional window, a lag window, \( w_{\text{lag}}(k) \) of size 2M (M<<N) is hence applied to the estimated autocorrelation sequence to improve the estimate (by reducing the variance of estimate),

\[
P_{BT}(e^{j\omega}) = \sum_{k=-M}^{M} r_x(k)w_{\text{lag}}(k)e^{-j\omega k}
\]  
(3-30)

The length \( M \) of the lag window is selected as the tradeoff between the variance and the bias of the estimate.

\[
\text{Var}\{P_{BT}(e^{j\omega})\} \approx P_s^2(e^{j\omega}) \frac{1}{N} \sum_{k=-M}^{M} w_{\text{lag}}^2(k), E(P_{BT}(e^{j\omega})) \approx \frac{1}{2\pi} P_s(e^{j\omega}) * W_{\text{lag}}(e^{j\omega})
\]  
(3-31)

Performance

The performance indexes for this family of methods are approximately the same: each method differs a bit in its resolution and variance, however, the overall performance defined by ‘the overall figure of merit’:

\[
M = \Delta \omega \cdot \text{Var}(\hat{P}_s(e^{j\omega})) \propto \frac{1}{N}
\]  
(3-32)

For all methods in this family are inversely proportional to \( N \), which implies that the performance of the methods in this family is fundamentally limited by the length of the data that is available.
3.6.2 Model-based Spectral Estimation for Broadband Signals

For broadband signals in noise, spectral estimation can be performed from using signal modeling, e.g. AR or ARMA models, to yield higher resolution spectra as compared to those from window-based methods. We shall consider signals with AR models in this part.

**AR Models**

In an AR(N) model, the data sample at time $n$ is assumed to be the weighted combination of the previous data samples $x(n-i), i=1,...,N$, in white noise $w(n)$,

$$x(n) = -\sum_{i=1}^{N} a_i x(n-i) + b_0 w(n)$$  \hspace{1cm} (3-33)

Where $N$ is the model order. This is equivalent to $x(n)$ being the output of an all-pole system $H(z)$ whose input is the white noise $w(n)$. The transfer function of the all pole system is

$$H_{AR}(z) = \frac{b_0}{1 + \sum_{i=1}^{N} a_i z^{-i}}$$  \hspace{1cm} (3-34)

To estimate the frequency spectrum of $x(n)$ thus becomes to estimate the model parameters $a_i$ under a selected criterion. For example, using the mean square criterion, model parameters $\{ a_i, i = 1, ..., N \}$ are estimated by solving a set of so-called normal equations

$$\sum_{m=1}^{N} a_m r_x(k,m) = -r_x(k,0); k = 1, ..., N$$  \hspace{1cm} (3-35)

Where $r_x(k,m) = E(x(n-k)x(n-m))$ is the data autocorrelation sequence, $k,m \geq 0$ and $x(n)$ is assumed to be real valued, and $E(.)$ denotes the expectation and is replaced by the summation under the ergodic data assumption. Once $\{ a_i \}$ are estimated, $b_0$ can be computed by solving.
\[
|b_0|^2 = r_x(0,0) + \sum_{m=1}^{N} a_m r_x(0,m)
\]  
(3-36)

The magnitude spectrum \( A(e^{j\omega}) \) and power spectrum \( P(e^{j\omega}) \) can be obtained by

\[
A(e^{j\omega}) = |H_{AR}(e^{j\omega})|^2 = \frac{b_0}{1 + \sum_{i=1}^{N} a_i e^{-j\omega}}
\]

\[
P(e^{j\omega}) = |H_{AR}(e^{j\omega})|^2
\]  
(3-37)

**The Modified Covariance Method**

The modified covariance method is similar to the covariance method in that no window is applied to the data. However, instead of finding the autoregressive model that minimizes the sum of the squares of the forward prediction error, the modified covariance method minimizes the sum of the squares of forward and backward prediction errors. As a result, the autoregressive parameters in the modified covariance method are found by solving a set of linear equations of the form given in (10).

\[
\begin{bmatrix}
  r_x(1,1) & r_x(2,1) & \cdots & r_x(p,1) \\
  r_x(1,2) & r_x(2,2) & \cdots & r_x(p,2) \\
  \vdots & \vdots & \ddots & \vdots \\
  r_x(1,p) & r_x(2,p) & \cdots & r_x(p,p)
\end{bmatrix}
\begin{bmatrix}
  a_p(1) \\
  a_p(2) \\
  \vdots \\
  a_p(p)
\end{bmatrix}
= 
\begin{bmatrix}
  r_x(0,1) \\
  r_x(0,2) \\
  \vdots \\
  r_x(0,p)
\end{bmatrix}
\]  
(3-38)

Where

\[
r_x(k,l) = \sum_{n-p}^{N-1} x(n-l)x^*(n-k) + x(n-p+l)x^*(n-p+k)
\]  
(3-39)

As with the covariance method, the autocorrelation matrix is not Toeplitz.
The Burg Algorithm

Similar to M-Covariance method, the Burg algorithm find a set of all pole model parameters that minimizes the sum of the squares of the forward and backward prediction errors. However in order to assure that the model is stable, this minimization is performed sequentially with respect to the reflection coefficients. Since the Burg algorithm does not apply a window to the data, the estimates of autocorrelation parameters are more accurate than those obtained with the autocorrelation method. In the analysis of sinusoids in noise, the Burg algorithm is subject to spectral line splitting and the peak locations are highly dependent upon the phases of sinusoids.