Spectral estimation

Cristian Iñiguez Rodriguez, 1566514@uab.cat

***Abstract*—This report presents an evaluation of various spectral estimation techniques and their practical uses.**

Introduction

The purpose of this report is to evaluate different spectral estimation methodologies and their applications. The first part is focused on the spectral estimation of a known Global Navigation Satellite System (GNSS) signal, and the performance of the periodogram. The same methodologie is applied to one of its variations, Blackman-Tukey. The part of the report is about the identification of two frequency components in a contaminated signal using both parametric and non-parametric spectral estimation techniques.

I. Answer to Question 1

The periodogram is a non-parametric estimator that provides an estimate of the power spectral density (PSD) of a signal without making assumptions about its structure. It can be calculated by following the equation

$$\hat{S}_x^{(per)}(e^{(j\omega)}) = \frac{1}{N}|\mathcal{F}\{x(n)\}|^2 \tag{1}$$

```
compute_periodogram = @(signal)
(1/length(signal)) * abs(fft(signal)).^2;
```

II. Answer to Question 2

GNSS signals can be distinguished because the spectrum is determined by the shape of the pulse that conforms the chip sequence.

GPS signals use BPSK(m) (binary phase shift keying) modulations. Assuming m = 1, its spectrum is given by the shape of the following pulse.

$$S_x(f)_{BPSK} = T_c sinc^2(fT_c) \tag{2}$$

Galileo signals, on the other hand, use BOC(m, n) (binary offset carrier) modulations. In practice, we will assume m = n = 1, giving the resulting expression of its spectrum.

$$S_x(f)_{BOC(1,1)} = T_c sinc^2(f\frac{T_c}{2})\sin^2 \pi f \frac{T_c}{2} \tag{3}$$

```
function plotPeriodogram(signal, mode)
N  = length(signal); Fs = 1e7;
f  = fftshift(linspace(-Fs/2, Fs/2, N));
figure;
if mode == "!."
    plot(f/Fs, 10*log10(signal));
else
    plot(f/Fs, 10*log10(signal), ".");
end

if (mode ≠ "" && mode ≠"!.")
hold on;
[¬, realS] = genTruePSD(Fs, N, mode, 0);
plot(f/Fs, 10*log10(realS),".");
hold off;
end
end
plotPeriodogram(compute_periodogram(...
                GNSS_signal_2), "");
```

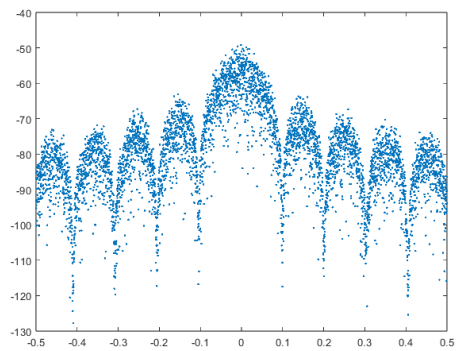

Fig. 1. Periodogram GPS signal (BPSK)

III. Answer to Question 3

```
plotPeriodogram(compute_periodogram( ...
                GNSS_signal_2), "GPS");
```

We can see in the following graph that Fig.1 has the shape of the spectrum of the GPS signal. It is probably not equal as the spectrum of the GPS signal because the GNSS signal has white noise added.

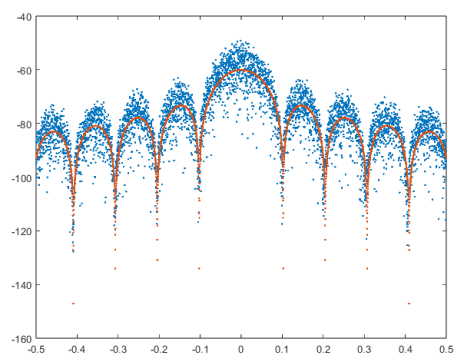

Fig. 2. GPS signal spectrum (orange) and periodogram (blue)

IV. Answer to Question 4

The periodogram is a biased estimator, whereas, when estimating the PSD, this one is distorted by the presence of the window in the form of a convolution with it.

$$E\{\hat{S}_x^{(per)}(e^{j\omega}) = W(e^{j\omega}) * S_x(e^{j\omega})\} \tag{4}$$

This convolution leads to two problems: spectral masking and power leakage. Spectral masking distorts the estimated PSD of the signal, while power leakage causes power to appear at frequencies where it does not actually exist. These effects result in a distorted spectrum and reduced resolution when viewing the spectrum along all frequency axes.

V. Answer to Question 5

When we are evaluating the periodogram, we have to keep in mind that it is a biased estimator, even though the bias disappears for a large number of samples (asymptotically unbiased).

```
function  bias_variance(compute_periodogram, ...
    mode)
Fs = 1e7; N  = 5000;
load Signals_PLAB3.mat GNSS_signal_1 ...
    GNSS_signal_2;

for n = 200:800:N
   interval = 1:n;
   [¬, signalR] = genTruePSD(Fs, n, mode, 0);
   f   = fftshift(linspace(-Fs/2, Fs/2, n));
    switch mode
        case 'GPS'
           signal = ...
    compute_periodogram(GNSS_signal_2(interval));
        case 'GAL'
           signal = ...
    compute_periodogram(GNSS_signal_1(interval));
    end
    figure;
    subplot(2,2,[3,4]);
    plot(fftshift(f), ...
        fftshift(10*log10(signal)));
    hold on;
    plot(fftshift(f), ...
        fftshift(10*log10(signalR)));
    title(['real Signal PSD :: '+mode+' vs ...
        GNSS Signal', '1 :'+string(n)]);
    hold off;

    subplot(2,2,1);
    plot(f, abs(signal-signalR));
    title(['absolute bias real Signal PSD :: ...
        '+mode+' vs GNSS Signal', '1 ...
        :'+string(n)]);

    subplot(2,2,2);
    plot(f, (signal-signalR).^2);
    title(['variance Signal PSD :: '+mode+' ...
        vs GNSS Signal', '1 :'+string(n)]);
end
end
```

VI. Answer to Question 6

To know which bandwidth to choose to select the main lobe, we see where it begins and ends, in mine it is in the interval [-1, 1] MHz. Then the necessary bandwidth is $20\mu s$

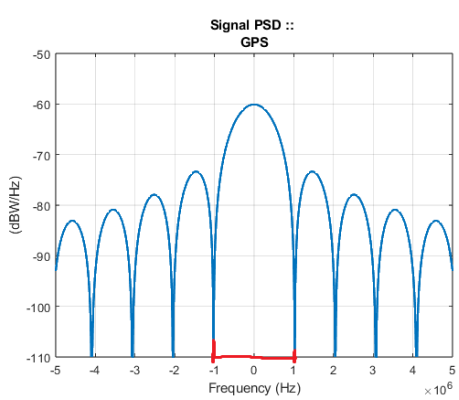

Fig. 3. Real GPS spectrum

VII. Answer to Question 7

This code of the function above calculates the variance of the signal.

```
subplot(2,2,2);
plot(f, (signal-signalR).^2);
title(['variance Signal PSD :: '+mode+' vs ...
    GNSS Signal', '1 :'+string(n)]);

bias_variance(compute_periodogram, "GPS");
```

The periodogram is not a consistent estimator because the variance does not decrease as the amount of processed data increases.

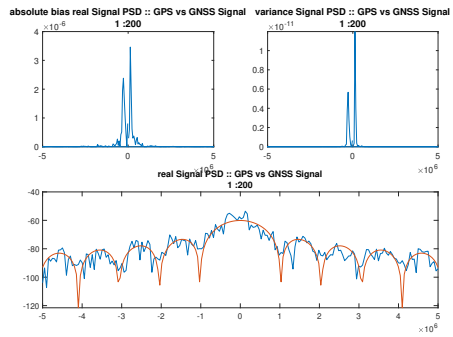

Fig. 4. bias, variance and comparation periodogram with 200 samples

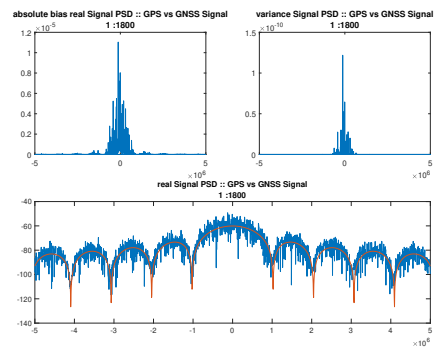

Fig. 5. bias, variance and comparation periodogram with 1800 samples

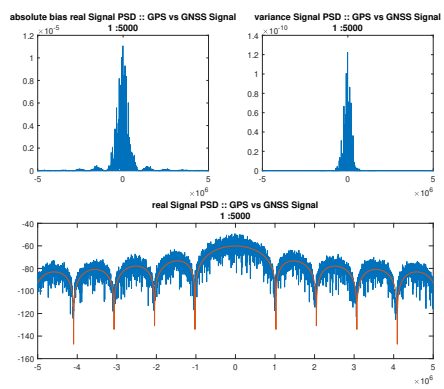

Fig. 6. bias, variance and comparation periodogram with 5000 samples

VIII. Answer to Question 8

The Blackman-Tukey method was developed with the idea of solving the problem of having samples of the autocorrelation that are badly estimated and better ones that are mixed at the time of calculating the original periodogram. This estimator is based on using a second window w'(m) on autocorrelation estimation in order to eliminate those samples that are poorly estimated as we move away from the correlation lag. The Blackman-Tukey estimator is defined by the following expression

$$\hat{S}_x^{BT}(e^{j\omega}) = \sum_{m=-M}^{M} w'(m)\hat{r}_x(m)e^{-j\omega m} = W'(e^{j\omega}) * \hat{S}_x(e^{j\omega}) \tag{5}$$

By applying a convolution with a second window to the periodogram, a low-pass filter is obtained that reduces the variance of the result, which is given by

$$var\{\hat{S}_x^{BT}(e^{j\omega})\} = S_x^{BT}(e^{j\omega})\frac{1}{N}\sum_{m=-M}^{M}[w'(m)]^2 = S_x^{BT}(e^{j\omega})\frac{E_{w'}}{N} \tag{6}$$

```
rxx = xcorr(GNSS_signal_2, "biased").';
N = length(rxx);
trian = @(N, n) [zeros(1, fix((N-n)/2)) ...
    triang(n)'];
tr  = trian(N, 221); tr(N) = 0;
plotPeriodogram(abs(fft(tr .* rxx)),"GPS")
```

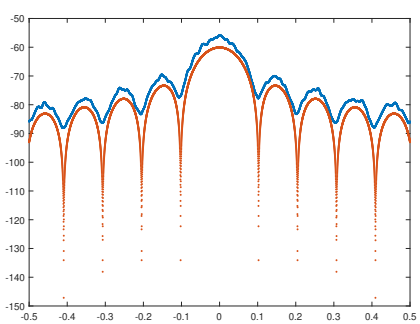

Fig. 7. Real (orange) and BT (blue), 221 samples window

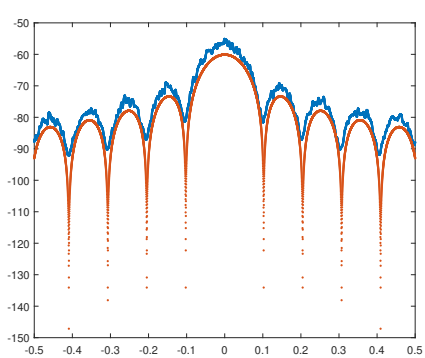

Fig. 8. Real (orange) and BT (blue), 555 samples window

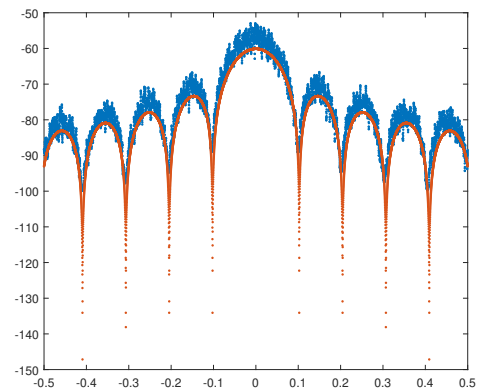

Fig. 9. Real (orange) and BT (blue), 3333 samples window

IX. Answer to Question 9

Blackman-Tuckey, like the periodogram, is a biased spectral estimator, but as the number of samples increases, the bias disappears (asymptotically unbiased), in terms of the consistency in the variance when adding that second window, what was sought and achieved is to eliminate that inconsistency. Therefore, with Blackman-Tuckey, a consistent and asymptotically unbiased spectral estimator is obtained

```
function BT_bias_variance(signal, mode)
Fs = 1e7;
trian = @(N, n) [zeros(1, fix((N-n)/2)) ...
    triang(n)'];
rxx = xcorr(signal, "biased").';

for n = 201:150:801
    clear tr signalR interval f;

    N   = length(rxx);
    tr  = trian(N, n);
    tr(N) = 0;
    tr = abs(fft(tr .* rxx));

    [¬, signalR] = genTruePSD(Fs, N, "GPS",0);
    signalR = signalR.';
    f   = fftshift(linspace(-Fs/2, Fs/2, N));

    figure;
    subplot(2,2,[3,4]);
    plot(fftshift(f), 10*log10(tr));
    hold on;
    plot(fftshift(f), 10*log10(signalR));
    title(['real Signal PSD :: '+mode+' vs ...
        GNSS Signal', '1 :'+string(n)]);
    hold off;

    subplot(2,2,1);
    plot(f, abs(tr-signalR));
    title(['absolute bias real Signal PSD :: ...
        '+mode+' vs GNSS Signal', '1 ...
        :'+string(n)]);

    subplot(2,2,2);
    plot(f, (tr-signalR).^2);
    title(['variance Signal PSD :: '+mode+' ...
        vs GNSS Signal', '1 :'+string(n)]);
end
end

BT_bias_variance(GNSS_signal_2, "GPS");
```

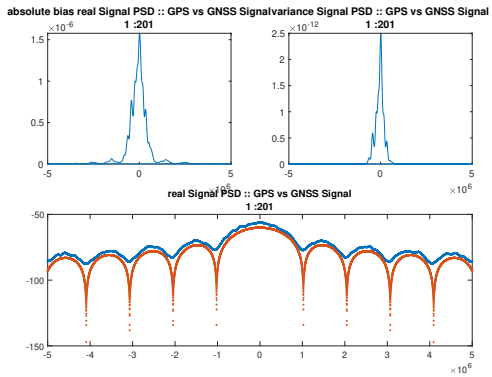

Fig. 10. bias, variance and comparation Blackman-Tukey with 201 samples

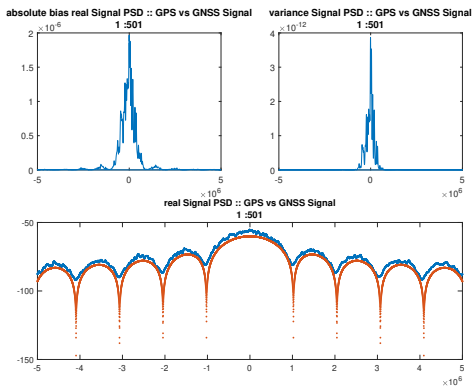

Fig. 11. bias, variance and comparation Blackman-Tukey with 501 samples

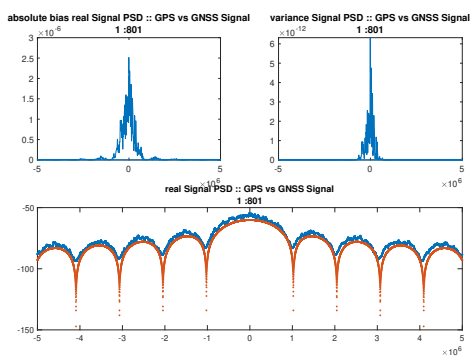

Fig. 12. bias, variance and comparation Blackman-Tukey with 801 samples

X. Answer to Question 10

Oone potential problem with using a periodogram for this purpose is that it can be difficult to accurately estimate the frequency spectrum of a signal, especially if the signal is noisy or non-stationary. Additionally, the results of a periodogram can be sensitive to the choice of windowing function, which can affect the accuracy of the estimated frequency spectrum.

```
function printPeaks(v, signal)
    str = @(x) string(x);
    compute_periodogram = @(signal) ...
        (1/length(signal)) * abs(fft(signal)).^2;
    Fs = 1e7;
    for i=1:length(v)
        s = compute_periodogram(signal(1:v(i)));

        [¬, locs] = findpeaks(s, 'SortStr', ...
            "descend");
        N = length(s);
        freqs = fftshift(linspace(-Fs/2, ...
            Fs/2-Fs/N, N))/Fs;

        disp(str(freqs(locs(1))) + " " + ...
            str(freqs(locs(2))) );
    end
end
plotting = [50 100 200 400];
printPeaks(plotting, Tone_signal_noisy);
```

The estimated frequencies for each case are as follows

time	Frequencies (Mhz)
$5\mu s$	[2.2, -3.4]
$10\mu s$	[2.1, -4.6]
$20\mu s$	[2.15, 2.05]
$40\mu s$	[2.15, 2.075]

XI. Answer to Question 11

With 200 and 400 samples, for the noise-free signal we obtain a very similar result. For 200 samples, the frequencies we obtain are as follows: 2.15 MHz and 2.05 MHz.

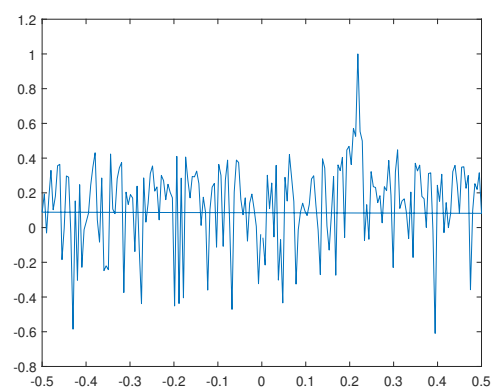

Fig. 13. Periodogram GPS signal with $20\mu s$

XII. Answer to Question 12

```
plotting = [50 100 200 400];
printPeaks(plotting, Tone_signal_noisy);
```

The number of samples chosen from the noisy signal is not enough for us to see the signal's frequencies using the periodogram. The frequencies it returns are neither the ones of interest nor the ones of interference. Even if we were able to obtain them, we would not be able to distinguish between them.

time	Frequencies (Mhz)
$5\mu s$	[1, -2.8]
$10\mu s$	[-3.6, -2.9]
$20\mu s$	[-3.65, 2.7]
$40\mu s$	[-3.75, 3.875]

XIII. Answer to Question 13

The concept of MUSIC is to utilize the orthogonality between the two subspaces to differentiate them. To achieve this goal, the first step is to diagonalize the autocorrelation matrix by extracting the eigenvectors and corresponding eigenvalues for each of them. The autocorrelation matrix is range-limited in the range $[-p+1, p-1]$, where p is an integer and $p \leq N$.

$$\begin{bmatrix} r_x(0) & ... & r_x(-p+1) \\ \vdots & \vdots & \vdots \\ r_x(p-1) & ... & r_x(0) \end{bmatrix} \tag{7}$$

Given that R_x it is a hermitian matrix, the decomposition into eigenvectors and eigenvalues is assured and takes the following form:

$$R_x = Q^H \Lambda Q \tag{8}$$

where $Q^H = Q^-1$ forms a base of orthonormal vectors placed in columns and Λ is a diagonal matrix containing the eigenvalues, $\Lambda = diag([\lambda_1, ..., \lambda_p])$

The eigenvectors that belong to the noise space have similar eigenvalues (equal to the noise power), while the eigenvectors associated with the signal space have eigenvalues that are the sum of the noise power and a part of the signal power. This allows us to differentiate between the two subspaces and indicates how many eigenvectors form the signal subspace. This number, called L, represents the number of frequency components in the signal and is provided to MUSIC. The noise vectors have indexes from $L+1$ to p, and the MUSIC estimator uses the orthogonality between the two subspaces through the following expression.

$$\hat{S}_x^{(MUSIC)}(e^{j\omega_k}) = \frac{1}{\sum_{n=L+1}^{p} |q_n^H s_k|^2} \tag{9}$$

where s_k is the frequency signature in $s \doteq [1, e^{j\omega}, ..., e^{j\omega(N-1)}]^T$ for a frequency ω_k determined that it will be necessary to sweep to obtain the complete spectrum $\hat{S}_x^{(MUSIC)}(e^{j\omega_k})$

```

function S_MUSIC = compute_MUSIC(signal, p, ...
    L, omega)
    N = length(signal);

    acorr = xcorr(signal, "biased");
    mid1  = acorr(N: N + p - 1);
    mid2  = acorr(N:-1:N - p + 1);
    rx    = toeplitz(mid1, mid2);

    [eigenvectors, ¬]  = svd(rx);
    eigen = eigenvectors(:, L+1:end);

    S_MUSIC = zeros(length(omega),1);
    for i = 1:length(omega)

    sk = exp(1i * omega(i) * (0:p-1)).';

    res = sk' * eigen * (eigen') * sk;
    S_MUSIC(i) = 1/abs(res);
    end
end
```

XIV. Answer to Question 14

Music method is said to be superresolution method because it performs better when you apply a much finer frequency sweep. The following table shows the estimated frequencies for a sweep at frequencies of 0.0002

time	Frequencies (Mhz)
$5\mu s$	[1.9728, 1.6428]
$10\mu s$	[1.9973, 1.9249]
$20\mu s$	[2.0093, 1.9524]
$40\mu s$	[2.0106, 1.9524]

MUSIC outperforms simple methods such as picking peaks of DFT spectra in the presence of noise, when the number of components is known in advance, because it exploits knowledge of this number to ignore the noise in its final report.

Unlike DFT, it is able to estimate frequencies with accuracy higher than one sample, because its estimation function can be evaluated for any frequency, not just those of DFT bins.

XV. Answer to Question 15

Since it is a superresolution method, the number of points we choose from the autocorrelation does not improve the search for frequency peaks. It is significantly more important to perform a better frequency sweep

```
Fs = 1e7;
v  = compute_MUSIC(Tone_signal_noisy, 200, ...
     2, 0.1:0.00002:2*pi);
f  = fftshift(linspace(-Fs/2, Fs/2, length(v)));
figure;
plot(f/Fs, v/max(v));
```

Since we only know that the signal is structured in the following way:

$$x(n) = A_s e^{j\omega_s n} + A_{int} e^{j\omega_{int} n} + w(n) \tag{10}$$

And music algorithm only provides us with where these peaks are, we cannot distinguish between the frequency of interest and the frequency of interference.

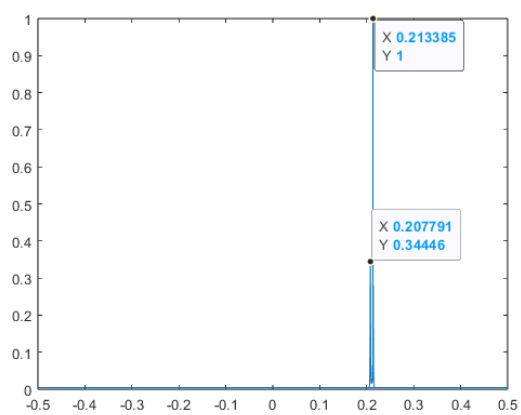

Fig. 14. MUSIC estimation with $20\mu s$

XVI. Answer to Question 16

The MUSIC (MUltiple SIgnal Classification) algorithm is a type of signal processing technique that is used to estimate the frequencies present in a signal. It is called a ”pseudo-spectrum” estimator because it uses a mathematical technique to find the peaks on the spectrum on the signal. So, as we can see, it only finds where the maximum peak is, but it does not give the spectrum.

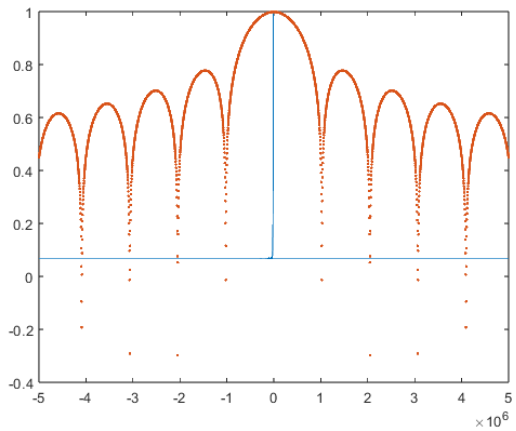

Fig. 15. Real GPS spectrum vs MUSIC estimation

Conclusion

We have studied the behavior of the periodogram for a signal with and without noise, we have seen how it is a biased spectral estimator, although asymptotically unbiased, and inconsistent. It is a biased estimator because it is as if a convolution were made with a window and the signal. To remove the inconsistency in the variance, we have applied the Blackman-Tukey method.

A pseudo-spectrum estimator such as Music has been compared with the periodogram to estimate two frequencies. We have seen how Music offers a better result with fewer samples, since it is also a superresolution estimator, but it is not able to show us the spectrum of the signal, only the frequencies of the peaks. In addition, in order to apply it, it is necessary to know how the signal is formed.