# Transformacje i funkcje statystyczne

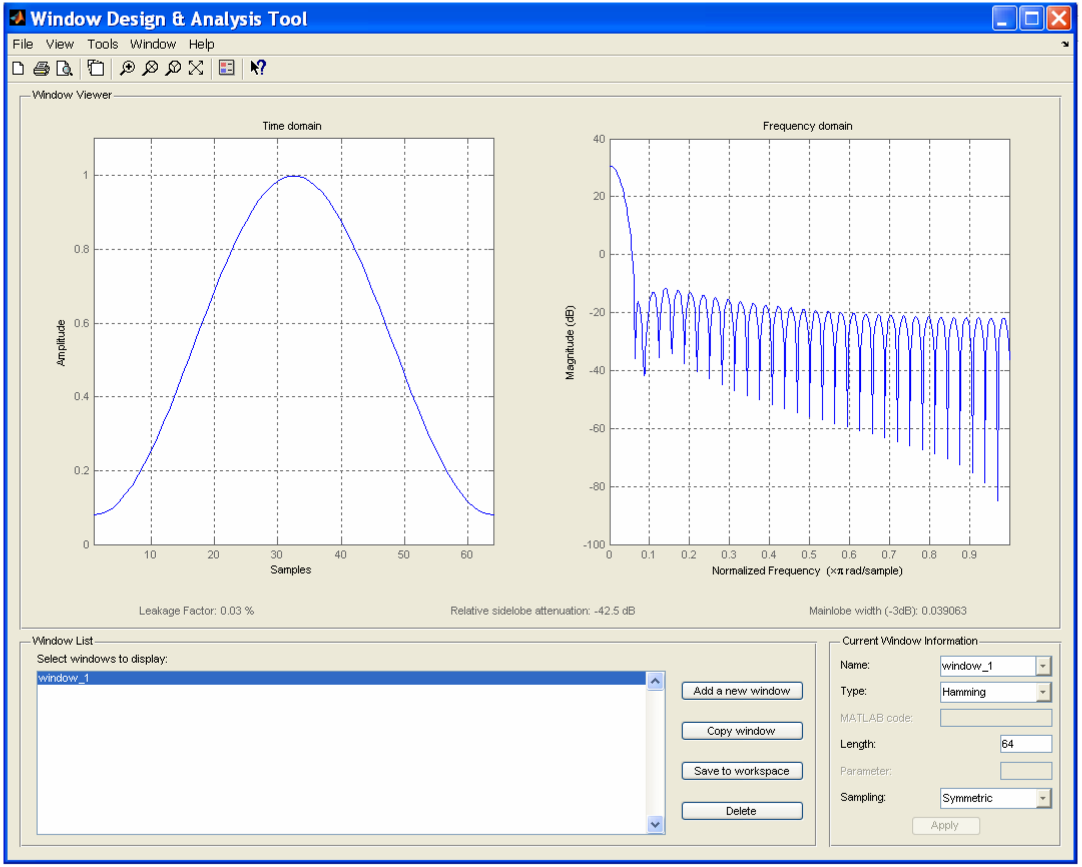
**Generacja okien:** win = window(@fwin,N);

**Generacja okien – gui:** wintool;

**Rodzaje niektórych okien:**

| | |
|---|---|
| @bartlett | - Bartletta. |
| @blackman | - Blackmana. |
| @chebwin | - Czebyszewa. |
| @gausswin | - gausowskie. |
| @hamming | - Hamminga. |
| @hann | - Hanna. |
| @kaiser | - Kaisera. |
| @rectwin | - prostokątne. |
| @triang | - trójkątne. |

# Transformacje i funkcje statystyczne




Działanie funkcji **wintool** (na przykładzie okna Hamminga)

# Transformacje i funkcje statystyczne

**Prosta i odwrotna jednowymiarowa dyskretna transformacja Fouriera:**

```
y = fft(x, nfft);
y = ifft(x, nfft);
```

**Prosta i odwrotna dwuwymiarowa dyskretna transformacja Fouriera:**

```
y = fft2(x, M, N);
y = ifft2(x, M, N);
```

# Transformacje i funkcje statystyczne

## Prosta i odwrotna jednowymiarowa dyskretna transformacja cosinusowa:

```
y = dct(x, ndct);
y = idct(x, ndct);
```

## Prosta i odwrotna dwuwymiarowa dyskretna transformacja cosinusowa:

```
y = dct2(x, ndct);
y = idct2(x, ndct);
```

# Transformacje i funkcje statystyczne

## Obliczanie widma przy użyciu algorytmu świergotowego (chirp Z- transform)

y = czt(x, m, w, a);

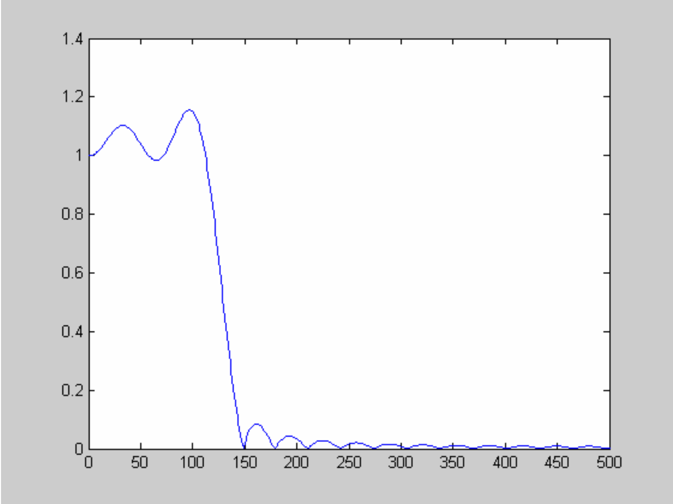
Algorytm świergotowy wykorzystuje m. in. się przy „powiększaniu” widma w wąskim paśmie częstotliwości [f1, f2], parametry w i a oblicza się następująco:

w = exp(-2*j*pi*(f2- f1))/(fs*m);

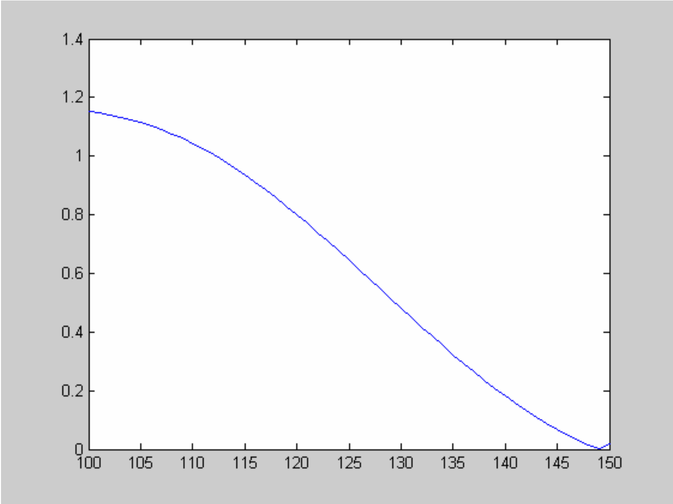
a = exp(2*j*pi*f1/fs);

# Transformacje i funkcje statystyczne

Widmo sygnału obliczone funkcją FFT




Fragment widma sygnału obliczony funkcją czt w zakresie f1 = 100 Hz, f2 = 150 Hz



# Transformacje i funkcje statystyczne

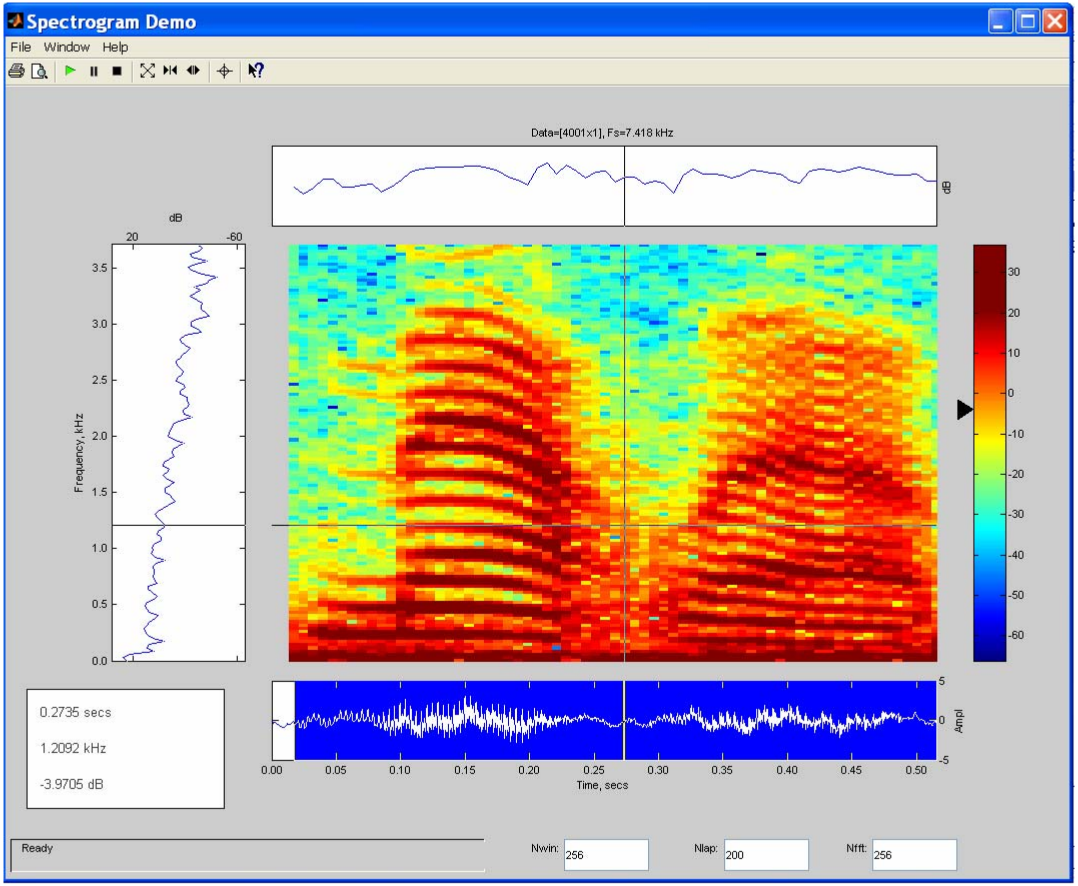
**Obliczanie spektrogramu sygnału jednowymiarowego:**

[y, f, t] = specgram(x, nfft, fs, hwin, novl)

Należy pamiętać, że novl < nfft oraz length(hwin) = nfft

**Spektrogram GUI wraz z odsłuchem sygnału:** specgramdemo(x, fs)

# Transformacje i funkcje statystyczne



# Transformacje i funkcje statystyczne

## Transformacja Hilberta przy wykorzystaniu FFT:

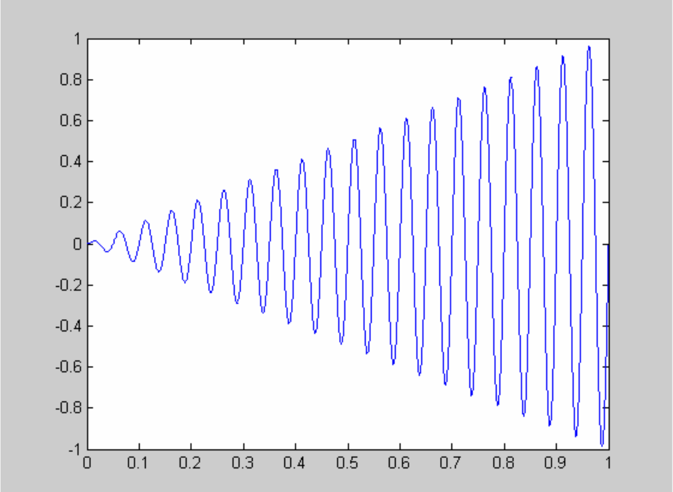
y = hilbert(x, nfft);

Działanie funkcji hilbert opiera się wyzerowaniu drugiego składnika sumy po prawej stronie poniższego równania  i pomnożeniu sumy przez 2

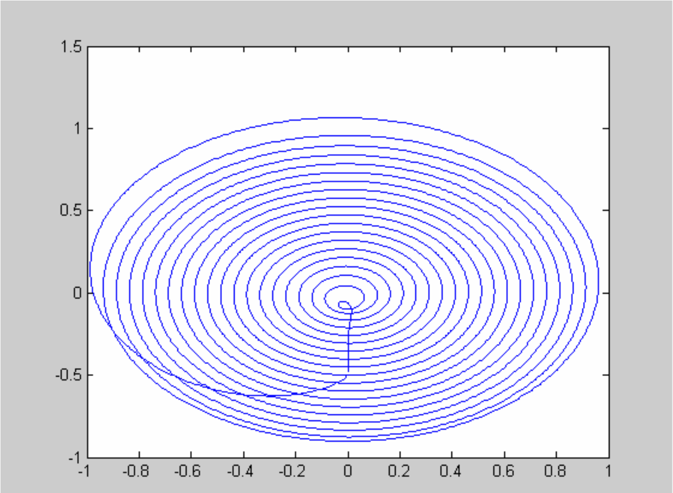
$$s(t) = A(t)\cos(\varphi(t)) = \frac{A(t)e^{j\varphi(t)}}{2} + \frac{A(t)e^{-j\varphi(t)}}{2}$$

# Transformacje i funkcje statystyczne

Przykład sygnału rzeczywistego




Rozszerzenie analitycznego powyższego sygnału rzeczywistego



# Transformacje i funkcje statystyczne

## Obliczanie cepstrum zespolonego

Definicja cepstrum zespolonego:

$$s^* = \mathrm{Re}(\mathrm{ifft}(\ln(\mathrm{fft}(\mathrm{s}))))$$

Ze względu na obecność widma fazowego, możliwa jest rekonstrukcja sygnału wejściowego

Obliczanie cepstrum zespolonego prostego i odwrotnego:

y = cceps(x);        y = icceps(x);

# Transformacje i funkcje statystyczne

## Obliczanie cepstrum rzeczywistego

Definicja cepstrum zespolonego:

$$s^{*} = \mathrm{Re}(\mathrm{ifft}(\ln(\mathrm{abs}(\mathrm{fft}(\mathrm{s})))))$$

Ze względu na brak widma fazowego, nie jest możliwa jest rekonstrukcja sygnału wejściowego

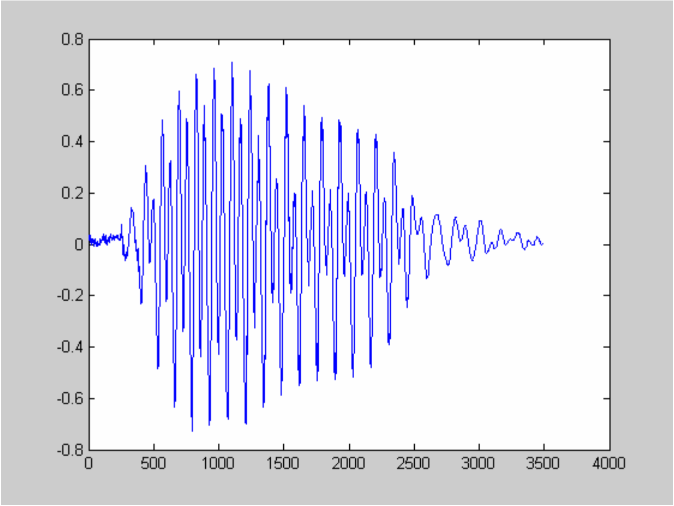
Obliczanie cepstrum rzeczywistego:

```
y = rceps(x);
```

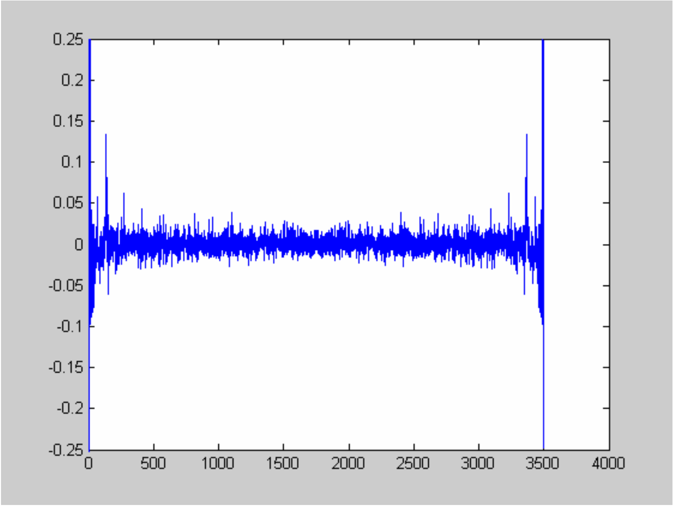
# Transformacje i funkcje statystyczne

Cepstrum może być postrzegane jako informacja o prędkości zmian w poszczególnych pasmach widma sygnału.

Obecnie stosowane jest przy analizowaniu sygnałów akustycznych, w szczególności ludzkiej mowy (głoski dźwięczne).




Fragment sygnału mowy




Współczynniki cepstralne

# Transformacje i funkcje statystyczne

## Estymacja mocy widma metodą periodogramu

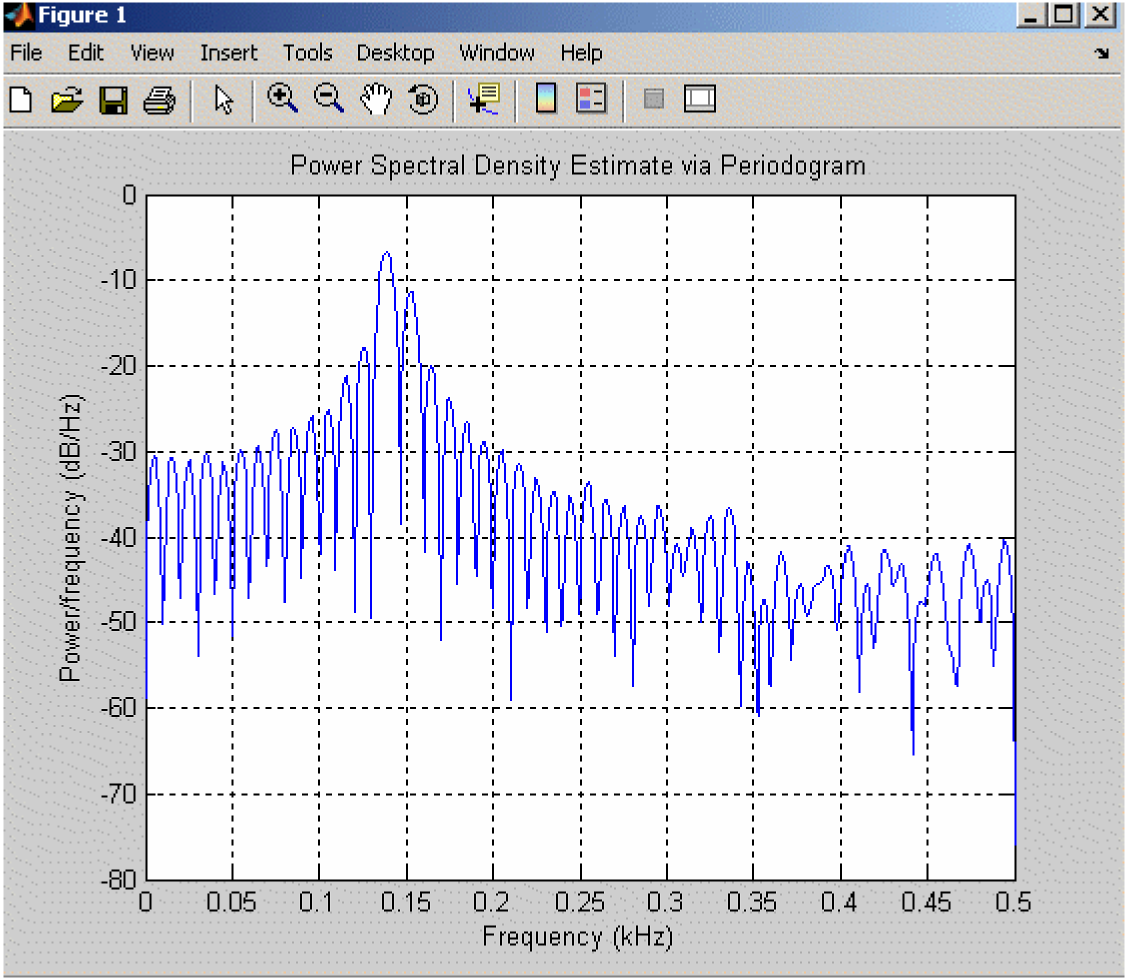
Periodogram sygnału wyznacza się zgodnie ze wzorem

$$S(\omega) = \frac{1}{N}\left(\sum_{n=1}^{N} s_n e^{-j\omega n}\right)^2$$

Obliczenie periodogramu sygnału:

```
[y, f] = periodogram(x, win, nfft, fs);
```

# Transformacje i funkcje statystyczne




Przykład periodogramu

# Transformacje i funkcje statystyczne

## Inne metody estymacji widma

### Estymacja mocy widma metodą Burga:

```
[y, W] = pburg(x, ord, nfft);
[y, f] = pburg(x, ord, nfft, fs);
```

### Estymacja mocy widma metodą Welsha:

```
[y, W] = pwelsh(x, ord, nfft);
[y, f] = pwelsh(x, ord, nfft, fs);
```

# Transformacje i funkcje statystyczne

**Estymacja mocy widma metodą kowariancji:**

[y, W] = pcov(x, ord, nfft);

[y, f] = pcov(x, ord, nfft, fs);

**Estymacja mocy widma zmodyfikowaną metodą kowariancji:**

[y, W] = pmcov(x, ord, nfft);

[y, f] = pmcov(x, ord, nfft, fs);

# Transformacje i funkcje statystyczne

**Estymacja mocy widma metodą Yule-Walkera:**

[y, W] = pyulear(x, ord, nfft);

[y, f] = pyulear (x, ord, nfft, fs);

**Estymacja widma korelacji wzajemnej:**

[z, W] = cpsd(x, y, hwin,noverlap, nfft);

[z, f] = cpsd(x, y, hwin,noverlap, nfft, fs);

# Transformacje i funkcje statystyczne

**Funkcje obliczeń statystycznych:**

```
y = mean(x);
y = var(x);
y = std(x);
z = cov(x, y);
z = corrcoef(x, y);
z = xcov(x, y);
z = xcorr(x, y);
z = xcorr2(x, y);
```