Wojciech Myszka

Laboratorium 2a: Interpolacja trygonometryczna wer. 39 z drobnymi modyfikacjami!

2018-10-24 10:44:41 +0200

Spis treści

1. Laboratorium 2a: Interpolacja trygonometryczna

1.1. Interpolacja trygonometryczna

W przypadku gdy funkcja (zjawisko), którą się zajmujemy jest okresowa czyli

$$g(y+t) = g(y)$$

dla wszystkich y do interpolacji najlepiej użyć funkcji okresowych. Dokonując zamiany zmiennych ($x = \frac{2\pi}{t}$) można rozpatrywać funkcje okresowe o okresie 2π.

Zadanie interpolacji trygonometrycznej polega na znalezieniu dla danej funkcji f:

$$t_n(x) = \sum_{j=0}^{n} c_j e^{ijx}$$

($i = \sqrt{-1}$). Oczekujemy, że wielomian ten przyjmie w $n+1$ punktach x_k z przedziału $[0, 2\pi]$ te same wartości co interpolowana funkcja, tzn.:

$$t_n(x_k) = f(x_k)$$

gdzie $x_k \neq x_l$ gdy $k \neq l$.

Zadanie te rozwiązuje się podobnie jak poprzednie rozwiązując układ $n+1$ równań liniowych z $n+1$ niewiadomymi $c_o, c_1, \ldots, c_n$

$$\sum_{j=0}^{n} c_j z_k^j = f(x_k)$$

gdzie $z_k = e^{ix_k}$.

Załóżmy, że węzły interpolacji są równoodległe czyli $x_k = \frac{2k\pi}{n+1}$ ($k = 0, 1, \ldots, n$).

Ze względu na to, że funkcje e^{ijx} są ortogonalne (w sensie iloczynu skalarnego) zagadnienie można potraktować jako rzut dowolnego wektora w przestrzeni na osie. Można pokazać, że współczynniki c_j można wyznaczyć w sposób następujący:

$$c_j = \frac{(f, e^{ijx})}{n+1} = \frac{1}{n+1} \sum_{k=0}^{n} f(x_k) e^{-ijx_k}$$

(zapis $(\bullet, \bullet)$ oznacza iloczyn skalarny.

Jak wiadomo, wielomian $t_n(x) = \sum_{j=0}^{n} c_j e^{ijx}$ przedstawiony może być w postaci alternatywnej:

$$t_n(x) = \frac{1}{2} a_0 + \sum_{j=1}^{m} (a_j \cos jx + b_j \sin jx) + \delta \frac{1}{2} a_{m+1} \cos(m+1)x$$

gdzie $\delta = 0$, a $m = \frac{1}{2}n$, gdy n parzyste oraz $\delta = 1$, a $m = \frac{1}{2}(n-1)$, gdy n nieparzyste. Współczynniki a_j i b_j równe są odpowiednio:

$$a_j = \frac{2}{n+1} \sum_{k=0}^{n} f(x_k) \cos jx_k$$

$$b_j = \frac{2}{n+1} \sum_{k=0}^{n} f(x_k) \sin jx_k$$

Powyższy wzór nie jest wykorzystywany do obliczeń. Zazwyczaj stosuje się Szybką Transformatę Fouriera (FFT). Algorytm opracowany przez Cooleya i Tookeya w 1965 pominę. Koszt podejścia klasycznego wymaga $(n+1)^2$ operacji. Natomiast, w przypadku algorytmu FFT, gdy $n+1$ może być przedstawione (rozłożone) jako iloczyn $r_1 r_2 \ldots r_p$ będzie rzędu $(n+1)(r_1 + r_2 + \cdots + r_p)$. Algorytm FFT najczęściej stosowany jest gdy $n + 1 = 2^k$, wymaga on wówczas $n \log_2 n$ działań.

Pewnym problemem stosowania interpolacji trygonometrycznej jest to, że funkcja jest okresowa. To znaczy zakładamy, że $t_n(0) = t_n(x_{n+1})$. Dokonując pomiarów zazwyczaj tak nie jest. W dalszym ciągu „można” stosować FFT, funkcja będzie okresowa, ale… Aby uwolnić się od tego problemu, nakłada się zmierzone dane funkcję okna. Typowa funkcja okna dla $x = 0$ i $x = 2\pi$ przyjmuje wartość zero co powoduje zniekształcenie badanego przebiegu.

1.2. Mathematica

1.2.1. Interpolacja trygonometryczna

Zajmę się głównie szybką transformatą Fouriera. Funkcja Fourier służy do wyliczenia FFT z zadanego przebiegu danych. Najprostsze jeż użycie będzie takie:

a = Fourier[{1, 1, 2, 2, 1, 1, 0, 0}]

$\{2.82843 + 0.i, -0.5 + 1.20711i, 0. + 0.i, 0.5 - 0.207107i, 0. + 0.i, 0.5 + 0.207107i, 0. + 0.i, -0.5 - 1.20711i\}$

Odwrotna transformata Fouriera:

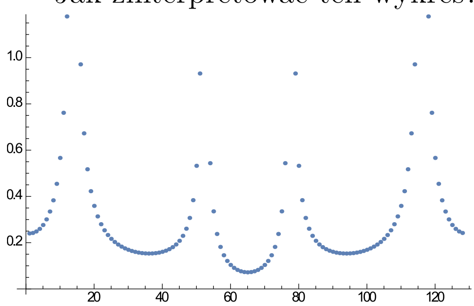
InverseFourier[a]

{1., 1., 2., 2., 1., 1., 0., −1.570092458683775*^-16} W każdym razie działa.

Teraz jakiś przebieg okresowy.

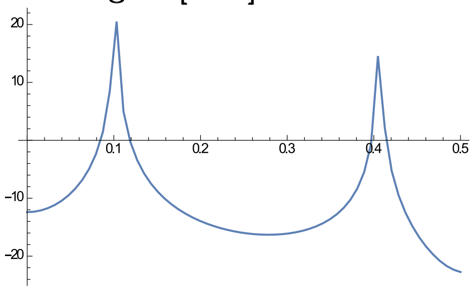
err = .0001;
data = Table[2Sin[0.2πn] + Sin[0.8πn] + RandomReal[{−err, err}], {n, 0, 127}];
fftdata = Fourier[data];
ListPlot[Abs[fftdata]]

Jak zinterpretować ten wykres? Czemu są cztery ekstrema (piki)?



Można również skorzystać z funkcji **Periodogram**.

Periodogram[data]



Jak zinterpretować ten rezultat?

Jak na podstawie powyższych wykresów zidentyfikować częstości przebiegu?

1. **Hint1**: Wygenerować jeden okres przebiegu o częstości 1 Hz i pokazać jego transformatę Fouriera oraz periodogram.
2. **Hint2**: Funkcja periodogram ma dodatkowy parametr **SampleRate**. Mówi on ile próbek na jednostkę czasu wykonano. Używany jest do skalowania osi X.

1.3. Matlab

1.3.1. Interpolacja trygonometryczna

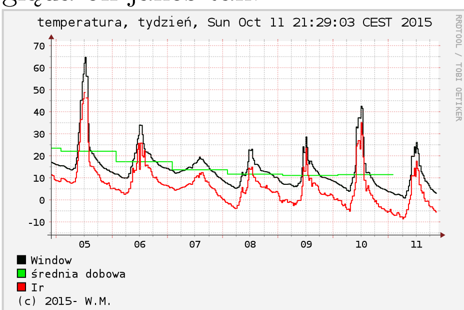
Podstawowym narzędziem jest funkcja **fft**:

```
Fs = 1000;                      % Sampling frequency
T = 1/Fs;                       % Sample time
L = 1000;                       % Length of signal
t = (0:L-1)*T;                  % Time vector
% Sum of a~50 Hz sinusoid and a~120 Hz sinusoid
x = 0.7*sin(2*pi*50*t) + sin(2*pi*120*t);
y = x + 2*randn(size(t));      % Sinusoids plus noise
plot(Fs*t(1:50),y(1:50))
title('Signal Corrupted with Zero-Mean Random Noise')
xlabel('time (milliseconds)')
NFFT = 2^nextpow2(L); % Next power of 2 from length of y
```

```
Y = fft(y,NFFT)/L;
f = Fs/2*linspace(0,1,NFFT/2+1);
% Plot single-sided amplitude spectrum.
plot(f,2*abs(Y(1:NFFT/2+1)))
title('Single-Sided Amplitude Spectrum of y(t)')
xlabel('Frequency (Hz)')
ylabel('|Y(f)|')
```

Zwracam uwagę, że pokazane rozwiązanie sugeruje, żeby długość ciągu danych była potęgą dwójki (polecenie: `NFFT = 2^nextpow2(L);`) czego nie wymaga Mathematica.

1.4. Zadania

1. Użyć interpolacji trygonometrycznej do znalezienia okresu zmienności tygodniowego (lub miesięcznego) wykresu temperatury. Wygląda on jakoś tak:


2. Przed rozpoczęciem tego zadania sugeruję zapoznanie się z informacjami zawartymi w rozdziale o Jupyterze oraz przeanalizowanie przykładowych notatników na temat wczytywania danych, eliminacji błędnych danych i szybkiej transformaty Fouriera.
3. Mając transformatę Fouriera warto spróbować przeprowadzić „naiwną, ręczną filtrację danych”, która polegać będzie na wyzerowani nieistotnych składowych transformaty i po zostawieniu tylko tych które wnoszą istotny wkład w przebieg oraz porównaniu przebiegów.

4. Opisać od czego zależy rozdzielczość (zdolność do identyfikowania w przebiegu złożonym składowych o bardzo bliskich częstotliwościach) szybkiej transformaty Fouriera. Można posiłkować się notatnikiem Jupyter.

1.5. Instrukcja w postaci jednego pliku…

…jest również dostępna.