OBLIGATORISK OPPGAVE 2

STA-2003-Tidsrekker

16. april 2019

Martin Soria RÃÿvang
Universitetet i TromsÃÿ

Inneholder 12 sider, inkludert forside.

Innhold

1 Oppgave

1.1 a

Har prosessen,

$$x(t) = x_1(t) + x_2(t) \tag{1}$$

,disse er *ukorrelerte* og *stasjonære* prosesser med forventning $E[x_1] = E[x_2] = 0$.
Vi har Wiener-Khinchin teoremene,

$$S_{xx}(f) = \mathscr{F}\{R_{xx}(\tau)\} = \int_{\infty}^{\infty} R_{xx}(\tau)e^{-i2\pi f\tau}\, d\tau \tag{2}$$

$$R_{xx}(\tau) = \mathscr{F}^{-1}\{S_{xx}(f)\} = \int_{\infty}^{\infty} S_{xx}(f)e^{i2\pi f\tau}\, df \tag{3}$$

Her er $\mathscr{F}$ Fourier transormasjon. Bruker at,

$$R_{xx}(\tau) = E[x_{t+\tau}x_t] \tag{4}$$

Løser vi denne med tanke pï£¡ ligning (1) fï£¡r vi,

$$R_{xx}(\tau) = E[(x_1(t+h) + x_2(t+h))(x_1(t) + x_2(t))] \tag{5}$$

Som gir,

$$R_{xx}(\tau) = E[x_1(t+h)x_1(t)] + E[x_1(t+h)x_2(t+h)] + E[x_1(t+h)x_2(t)] + E[x_2(t+h)x_2(t)] \tag{6}$$

Siden prosessene er ukorrelerte vil man kunne separere forventingene, $E[x_1(t+h)]E[x_2(t+h)]$
Dette gir da,

$$R_{xx}(\tau) = E[x_1(t+h)x_1(t)] + E[x_1(t+h)]E[x_2(t+h)] + E[x_1(t+h)]E[x_2(t)] + E[x_2(t+h)x_2(t)] \tag{7}$$

Dette kan separeres slik at,

$$R_{xx}(\tau) = E[x_1(t+h)x_1(t)] + E[x_1(t+h)]E[x_2(t+h)] + E[x_1(t+h)]E[x_2(t)] + E[x_2(t+h)x_2(t)] \tag{8}$$

Siden $R_{x_1x_1}(\tau) = E[x_1(t+\tau)x_1(t)]$ fï£¡r vi,

$$R_{xx}(\tau) = R_{x_1x_1}(\tau) + E[x_1(t+h)]E[x_2(t+h)] + E[x_1(t+h)]E[x_2(t)] + R_{x_2x_2}(\tau) \tag{9}$$

Vet at forventningen til prosessene er null, $E[x_1(t+h)]E[x_2(t+h)] = 0$ og $E[x_1(t+h)]E[x_2(t)]$,

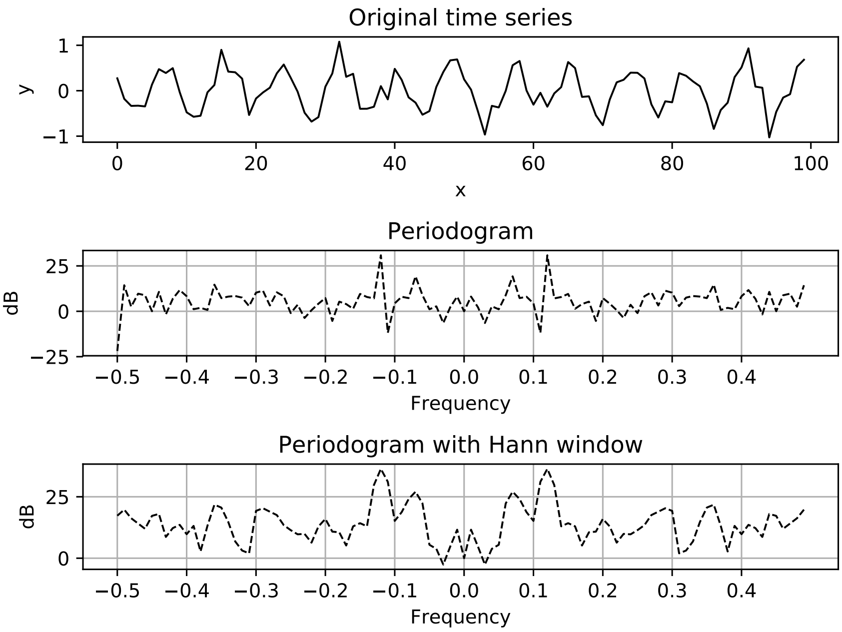
$$R_{xx}(\tau) = R_{x_1x_1}(\tau) + R_{x_2x_2}(\tau) \tag{10}$$

Siden vi har summer kan vi dele integralet opp i sum,

$$S_{xx}(f) = \underbrace{\int_{\infty}^{\infty} R_{x_1x_1}(\tau)e^{-i2\pi f\tau}\, d\tau}_{=S_{x_1x_1}(f)} + \underbrace{\int_{\infty}^{\infty} R_{x_2x_2}(\tau)e^{-i2\pi f\tau}\, d\tau}_{=S_{x_2x_2}(f)} \tag{11}$$

1.2 b

I figur(1) har vi plot av tidsrekken, periodogrammet og tilslutt periodogram med hann vindu.



Figur 1: Beregnet periodogram av tidsrekken. Man kan se at vindu vil hjelpe med ï£¡ gi mer tydelig seperasjon mellom frekvensene, hvis man ser pï£¡ rundt frekvens 0.3-0.4 kan man tydelig se stor forskjell. Vindu er svï£¡rt hjelpsomt hvis man har kraftig signal med et underliggende svakt signal man vil oppdage.

Her ser man at hann vinduet gir en glattere kurve, og lar ikke frekvensene lekke over til de andre frekvensene like mye som i det vanlige periodogrammet. Frekvensene ser ut til ï£¡ vï£¡re $f = 0.12$ og $f = 0.07$.

1.3 c

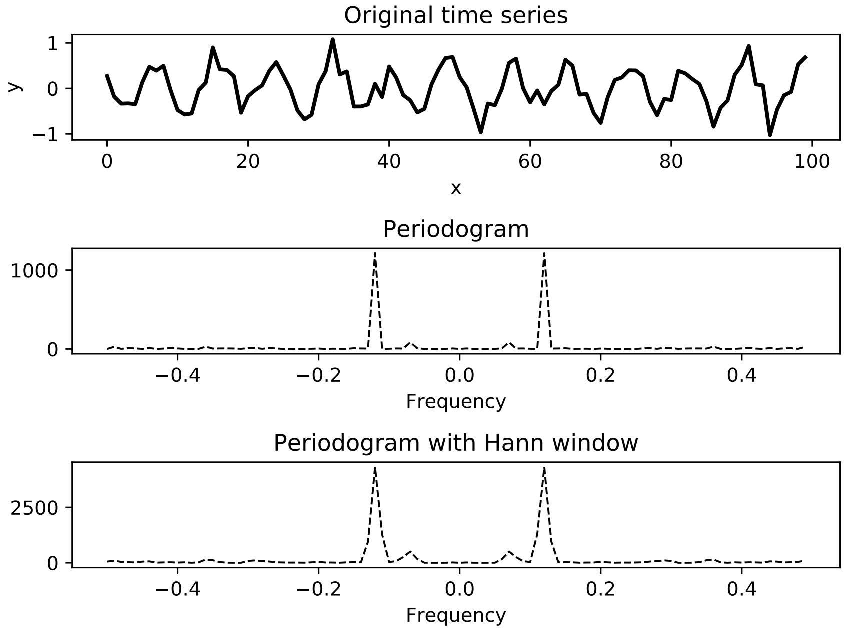
Som nevnt i oppgave b sï£¡ fjerner vinduet mer av lekasjen i frekvensspekteret, fordi den har mindre sidelober enn den idelle w[k] = 1, men dette vil ï£¡ke bredden pï£¡ hovedloben slik at man fï£¡r litt dï£¡rligere opplï£¡sning. I figure(2) har jeg ikke brukt dB skala slik at differansen er stï£¡rre mellom styrken pï£¡ frekvensene. Her kan man tydelig se at vinduet hjelper med ï£¡ se svakere periodisitetet i spectrumet. Periodogram ble laget ved bruk av Fourier transform som vist i ligning(12).

$$S(f) = \frac{1}{N}|\mathscr{F}\{x[n]\}|^2 \tag{12}$$

der $\mathscr{F}$ er DTFT, *Discrete Time Fourier Transform* og N er antall datapunkter. Ved bruk av vindu har vi ligning(13).

$$S_w(f) = \frac{1}{NU}|\mathscr{F}\{w[n]x[n]\}|^2 \tag{13}$$

Her er $w[n]$ vinduet og U er normaliseringskonstant for ï£¡ fjerne energien som blir lagt til prosessen fra vinduet, i dette tilfelle er w[n] *Hann window*. Vi bruker vindu for ï£¡ minske spektral lekasje som kommer av at vi ikke kan ha en *ekte* fourier transform som krever at signal vi har er uendelig langt. Derfor vil vi alltid ha en ende pï£¡ signal som man antar er null. Dette betyr at vi bruker et ideelt vindu med lengde M(signal lengden) nï£¡r vi tar Fourier transform. Nï£¡r man ganger inn tidsdomenet sï£¡ konvolverer man i frekvensdomenet, og siden det ideele vinduet har store sidelober medfï£¡rer dette til at man fï£¡r ut nï£¡rliggende frekvenser i hverandre og dermed gir spektrallekasje. Ved bruk av vindu som Hann vindu sï£¡ kan man redusere dette siden de har smalere sidelober.

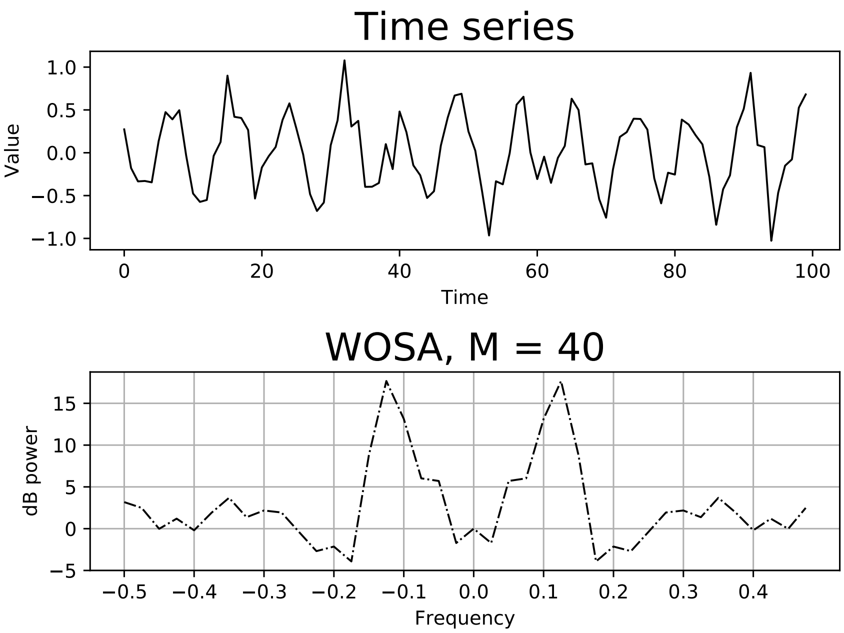


Figur 2: Beregnet periodogram av tidsrekken uten dB skala.

1.4 d

Ved ï£¡ implementere WOSA (Weighted/Welchs overlapped segment averaging) pï£¡ tidsrekken fikk vi resultatet vist i figur(3).

Her ble hvert segment vektet som gir et kraftigere utslag der det er sterke periodisiteter i forhold til resten av spekteret. Her ser man at frekvensene pï£¡ $f = 0.12$ og $f = 0.1$ er ganske kraftige i forhold til alle de andre. Man kan ogsï£¡ se at frekvensene er mye tydeligere i WOSA enn i periodogrammet og hann vinduet i dB-skala som vist i figur(1).

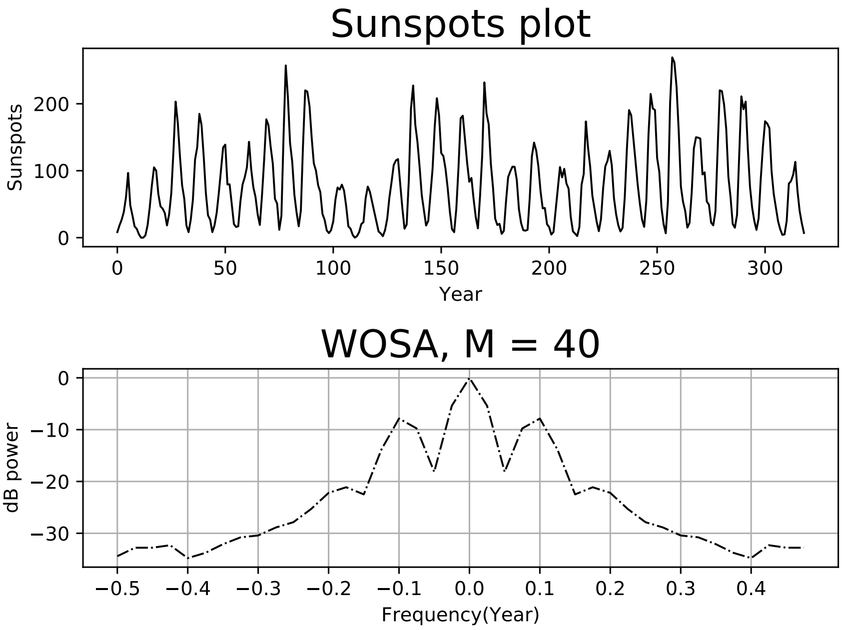


Figur 3: WOSA med 50% overlapp, N = 40

2 2

2.1 a-b

Her plottet vi sol-dataen som vist i ï£¡verste vindu i figur(4).



Figur 4: Soldata, WOSA med 50% overlapp, N = 40. Ser ut som man har en kraftig periode pï£¡ 10 ï£¡r.

Her ble det brukt WOSA med 50% overlapp og vindu-stï£¡rrelse $N = 40$. Her ser man sterk DC($f = 0$) og en frekvens pï£¡ rundt 0.1, $f \approx 0.1$. Dette gir en periode pï£¡ $T \approx 10$, noe som ser ut til ï£¡ stemme hvis man teller antall topper som er ≈ 10 per 100, altsï£¡ periode pï£¡ ti ï£¡r.

3 Appendix

```
import numpy as np
import matplotlib.pyplot as plt


data = np.genfromtxt('tidsrekke_oblig2_oppg1.txt')

def find_freqs(spectrum, freq, freqs = 5):
    """Find the frequency with highest power"""
    frequencies = []
    # Slice away double side
    spectrum = spectrum[int(len(spectrum)/2):len(spectrum)]
    freq = freq[int(len(freq)/2):len(freq)]
    # Find the most powerfull frequencies
    for i in range(freqs):
        idx = np.argmax(spectrum)
        frequencies.append(freq[idx])
        spectrum = np.delete(spectrum,idx)
 return frequencies


def periodogram(x, dt):
    """Regular periodogram"""
    N = len(x)
    spectrum = np.abs(np.fft.fftshift(np.fft.fft(x))**2)
    spectrum *= dt/ N
    freq = np.fft.fftshift(np.fft.fftfreq(N, dt))

    return freq, spectrum


def w_periodogram(x, dt):
    """Windowed periodogram"""
    N = len(x)
    n = np.arange(0,N,1)
    # Hann window
    window = (1/2)*(1 - np.cos(2*np.pi*n/(N-1)))
    U = (1/N)*np.sum(window**2)
    spectrum = np.abs(np.fft.fftshift(np.fft.fft(window*x)))**2
    spectrum *= (dt/(N*U))
    freq = np.fft.fftshift(np.fft.fftfreq(N, dt))
    return freq, spectrum

dt = 1
freq, spectrum = periodogram(data, dt)
freqw, wspectrum = w_periodogram(data, dt)

# Find index corresponding to f = 0
idx = np.where(freq == 0)
widx = np.where(freqw == 0)

# Plot
fig, ax = plt.subplots(3,1)
ax[0].plot(data, color = 'black', linewidth = 1)
ax[0].set_title('Original time series')
ax[0].set_xlabel('x')
ax[0].set_ylabel('y')
ax[1].plot(freq, 10*np.log10(spectrum/spectrum[idx]),'--', color = 'black', linewidth = 1)
ax[1].set_title('Periodogram')
ax[1].set_xlabel('Frequency')
ax[1].set_ylabel('dB')
ax[2].plot(freqw, 10*np.log10(wspectrum/wspectrum[widx]),'--', color = 'black', linewidth = 1)
ax[2].set_title('Periodogram with Hann window')
ax[2].set_xlabel('Frequency')
ax[2].set_ylabel('dB')
ax[1].set_xticks([x for x in np.arange(-0.5,0.5,0.1)])
ax[2].set_xticks([x for x in np.arange(-0.5,0.5,0.1)])
ax[1].grid()
ax[2].grid()
plt.tight_layout()
plt.savefig('rapport/taskb.pdf', bbox_inches = 'tight',
    pad_inches = 0)
plt.show()
```

```python
import matplotlib.pyplot as plt
import numpy as np

# Load file
data = np.genfromtxt('tidsrekke_oblig2_oppg1.txt')


def WOSA(x, M, dt = 1):
    """Implementation of WOSA"""
    n = np.arange(0, M, 1)
    window = (1/2)*(1 - np.cos(2*np.pi*n/(M-1)))
    U = (1/M)*np.sum(window**2)
    spectrum = np.zeros(M)
    n_windows = 2*int(len(x)/(M-1))
    for i in range(n_windows):

        # Start with window 0-40
        if i == 0:
            spectrum_temp = np.fft.fftshift(np.fft.fft(window*x[0:40]))
            plt.plot(x[0:40])
            t = np.arange(0, 40, 1)
            plt.plot(t, window)


        # Start overlapping
        else:
            spectrum_temp = np.fft.fftshift(np.fft.fft(window*x[(i*int(M/2)):(i+2)*int(M/2)]))
            plt.plot(x[0:(i+2)*int(M/2)])
            t = np.arange((i*int(M/2)), (i+2)*int(M/2), 1)
            plt.plot(t, window)
        spectrum += (dt/(M*U))*np.abs(spectrum_temp)**2
    spectrum /= n_windows
    plt.show()
    freq = np.fft.fftshift(np.fft.fftfreq(M, dt))
    print('Number of windows: %s'%n_windows)
    return freq, spectrum

M = 40
dt = 1
freq, spectrum = WOSA(data, M)
# Find index corresponding to f = 0
idx = np.where(freq == 0)

# Plot
fig, ax = plt.subplots(2,1)
ax[0].plot(data, linewidth = '1', color = 'black')
ax[0].set_title('Time series', fontsize = '20')
ax[0].set_ylabel('Value')
ax[0].set_xlabel('Time')
ax[1].plot(freq, 10*np.log10(spectrum/spectrum[idx]), '-.', linewidth = '1', color = 'black')
ax[1].set_title('WOSA, M = %s'%M, fontsize = '20')
ax[1].set_xlabel('Frequency')
ax[1].set_ylabel('dB power')
ax[1].set_xticks([x for x in np.arange(-0.5,0.5,0.1)])
ax[1].grid()
plt.tight_layout()
plt.savefig('rapport/taskd.pdf', bbox_inches = 'tight',
    pad_inches = 0)
plt.show()
```

Figur 6: Task D

```python
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

filedir = os.path.dirname(__file__)
filename = 'SN_y_tot_V2.0.txt'

file = os.path.join(filedir, filename)

# load file
datatable = pd.read_csv(file,sep='\t',header=None,engine='python')
time = np.zeros(len(datatable))
sunspots = np.zeros(len(datatable))

for i in range(len(datatable)):
    time[i] = datatable.values[i,0][0:6]
    sunspots[i] = datatable.values[i,0][8:13]


def WOSA(x, M, dt = 1):
    """Implementation of WOSA"""
    n = np.arange(0, M, 1)
    window = (1/2)*(1 - np.cos(2*np.pi*n/(M-1)))
    U = (1/M)*np.sum(window**2)
    spectrum = np.zeros(M)
    n_windows = 2*int(len(x)/(M-1))-1
    x = np.pad(x, (0, M), 'constant')
    for i in range(n_windows):
        # Start with window 0-40
        if i == 0:
            spectrum_temp = np.fft.fftshift(np.fft.fft(window*x[0:40]))
            plt.plot(x[0:40])
            t = np.arange(0, 40, 1)
            plt.plot(t, 200*window)

        # Start overlapping
        else:
            spectrum_temp = np.fft.fftshift(np.fft.fft(window*x[(i*int(M/2)):(i+2)*int(M/2)]))
            plt.plot(x[0:(i+2)*int(M/2)])
            t = np.arange((i*int(M/2)), (i+2)*int(M/2), 1)
            plt.plot(t, 200*window)
        spectrum += (dt/(M*U))*np.abs(spectrum_temp)**2
    spectrum /= n_windows
    plt.show()
    plt.close()
    freq = np.fft.fftshift(np.fft.fftfreq(M, dt))
    print('Number of windows: %s'%n_windows)
    return freq, spectrum

M = 40
dt = 1
freq, spectrum = WOSA(sunspots, M)
# Find index corresponding to f = 0
idx = np.where(freq == 0)

# plot
fig, ax = plt.subplots(2,1)
ax[0].plot(sunspots, linewidth = '3', color = 'black')
ax[0].set_title('Sunspots plot', fontsize = '20')
ax[0].set_ylabel('Sunspots')
ax[0].set_xlabel('Year')
ax[1].plot(freq, 10*np.log10(spectrum/spectrum[idx]), '-.', linewidth = '1', color = 'black')
ax[1].set_title('WOSA, M = %s'%M, fontsize = '20')
ax[1].set_xlabel('Frequency')
ax[1].set_ylabel('dB power')
ax[1].set_xticks([x for x in np.arange(-0.5,0.5,0.1)])
ax[1].grid()
plt.tight_layout()
plt.savefig('rapport/task2.pdf', bbox_inches = 'tight',
    pad_inches = 0)
plt.show()
```

4 Referanser