OBLIGATORISK OPPGAVE 2

STA-2003-Tidsrekker

18. mars 2019

Martin Soria RÃÿvang
Universitetet i TromsÃÿ

Inneholder 11 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 theoremene,

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

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

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

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

Løser vi denne med tanke på ligning (1) får 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 $R_{x_1x_1}(\tau) = E[x_1(t+\tau)x_1(t)]$ får 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 $E[x_1(t+h)]E[x_2(t)]$,

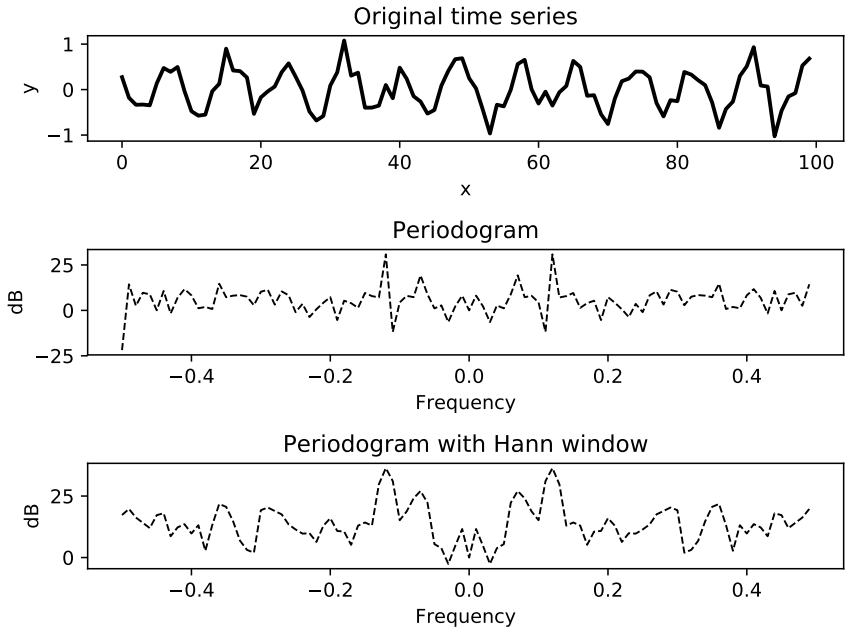
$$R_{xx}(\tau) = R_{x_1x_1}(\tau) + R_{x_2x_2}(\tau) \tag{10}$$

Siden vi har summer kan vi delle 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) ser vi tidsrekken i øverste vindu, periodogram, og periodogram med hann vindu.



Figur 1: Beregnet periodogram av tidsrekken.

Her ser man at hann vinduet gir en glattere kurve, og lar ikke frekvense lekker over like mye som i det vanlige periodogrammet. Frekvensene ser ut til å være rundt 0.1 og 0.05.

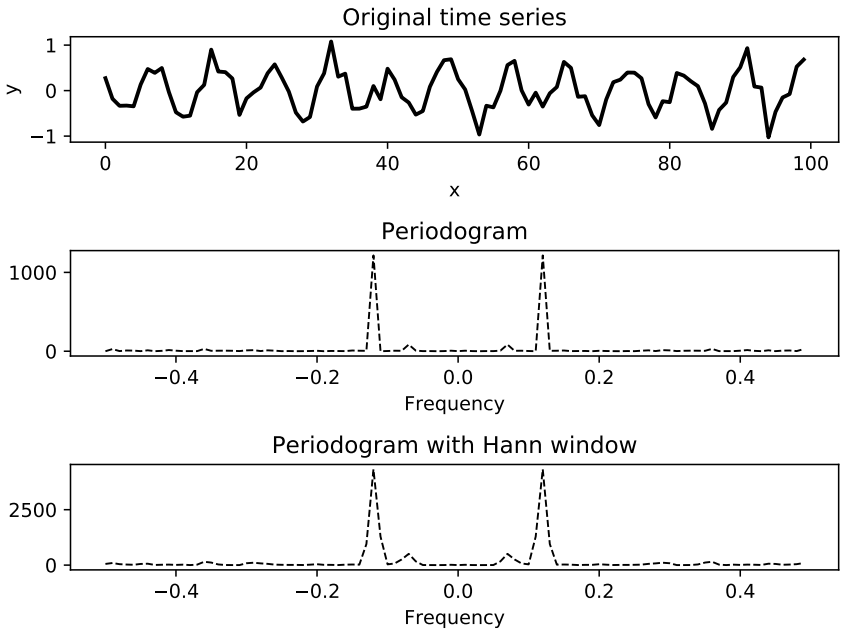
1.3 c

Som nevnt i oppgave b så fjerner vinduet mer av lekasje i frekvensspekteret, grunnet at den har lavere sidelober enn den idelle w[k] = 1, men dette vil øke bredden på hovedloben slik at man får litt dårligere oppløsning. I figure(2) har jeg ikke brukt dB skala slik at differansen er større mellom styrken på frekvensene. Her kan man tydelig se at vinduet hjelper med å se svakere periodisitetet i spectrumet.

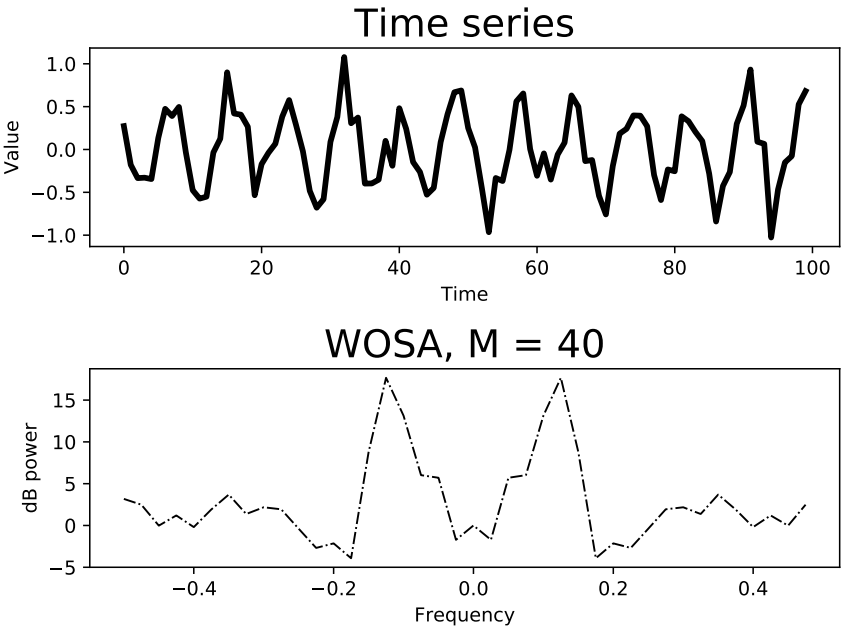
1.4 d

Ved å implementere WOSA (Weighted/Welchs overlapped segment averaging) på tidsrekken fikk man resultatet vist i figur(3).

Her ble hver av segment vektet som gir et kraftigere utslag der det var sterke periodisiteter i forhold til resten av spekteret. Her ser man at frekvensen ≈ 0.1 er svært kraftig og ≈ 0.05 viser seg også ganske kraftig i forhold til alle de andre. Man kan se at frekvensene er mye tydligere i WOSA enn i periodogrammet og hann vinduet i dB skala som vist i figur(1).



Figur 2: Beregnet periodogram av tidsrekken uten dB skala.

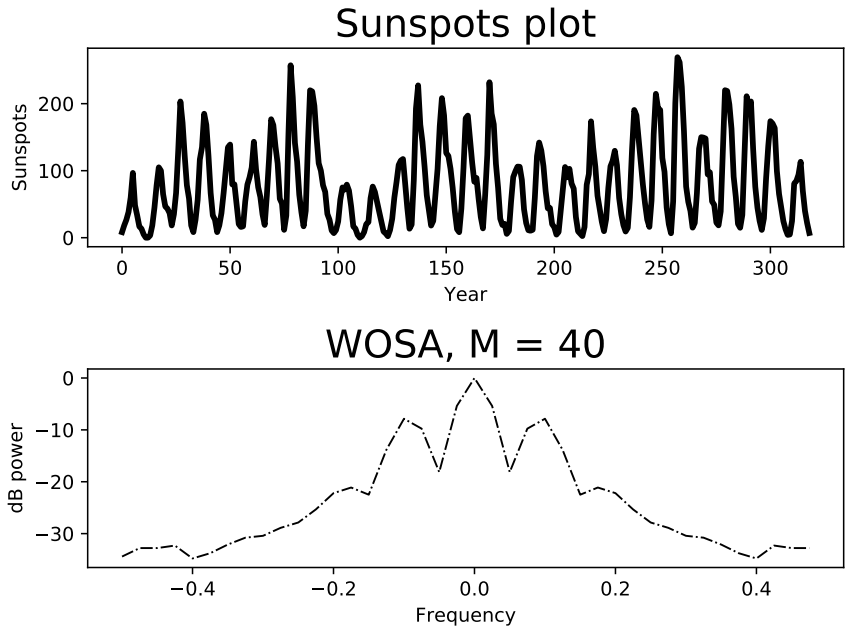


Figur 3: WOSA med 50% overlapp, N = 40

2 2

2.1 a-b

Her plottet vi sol-dataen som vist i øverste vindu i figur(4).



Figur 4: Soldata, WOSA med 50% overlapp, N = 40

Her ble det brukt WOSA med 50% overlapp og vindu-størrelse $N = 40$. Her ser man sterk DC og en frekvens på rundt 0.1, $f \approx 0.1$. Dette gir en periode på $T \approx 10$, noe som ser ut til å stemme hvis man teller antall topper som er ≈ 9 per 100.

3 Appendix

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


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


        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 = 2)
        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')
        plt.tight_layout()
        plt.savefig('rapport/taskb.pdf', bbox_inches = 'tight',
            pad_inches = 0)
        plt.show()


```

Figur 5: Task 1B-C

```
        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 = '3', 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_xlim([0,1/(2*dt)])
        plt.tight_layout()
        plt.savefig('rapport/taskd.pdf', bbox_inches = 'tight',
            pad_inches = 0)
        plt.show()

```

Figur 6: Task D

```
        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
            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_xlim([0,1/(2*dt)])
        plt.tight_layout()
        plt.savefig('rapport/task2.pdf', bbox_inches = 'tight',
             pad_inches = 0)
        plt.show()

```


Figur 7: Task 2 code

4 Referanser