OBLIGATORISK OPPGAVE 3

STA-2003-Tidsrekker

6. mai 2019

Martin Soria Røvang
Universitetet i Tromsø

Inneholder 13 sider, inkludert forside.

Innhold

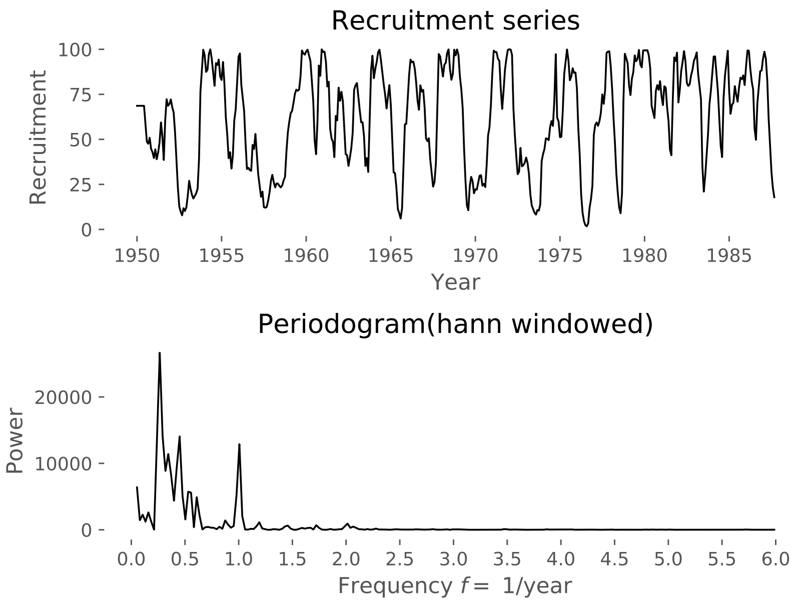
1 Oppgave

1.1 a)

Ved bruk av en vektet periodogram vist i ligning(1.1) kan vi se på energien/tid til tidsrekken. Denne viser at vi har kraftig periode rundt $f = 0.5/år$ og på $f = 1/år$(periode på ett år $T = 1år$), derfor kan man se at det er sesongvariasjoner i tidsrekken.

$$S_{xx}(f) = \frac{\Delta t}{NU}\left|\mathfrak{F}\left\{w[n]\cdot x[n]\right\}\right|^2 \tag{1.1}$$

Her er S_{xx} kraften på frekvenskomponentene, Δt er tidssteget (i dette tilfelle $\Delta t = 1$), $\mathfrak{F}$ er Fouriertransformasjonen, $w[n]$ er vinduet (brukt hann vindu) og $x[n]$ er tidsrekken. Resultatet er i figur(1).



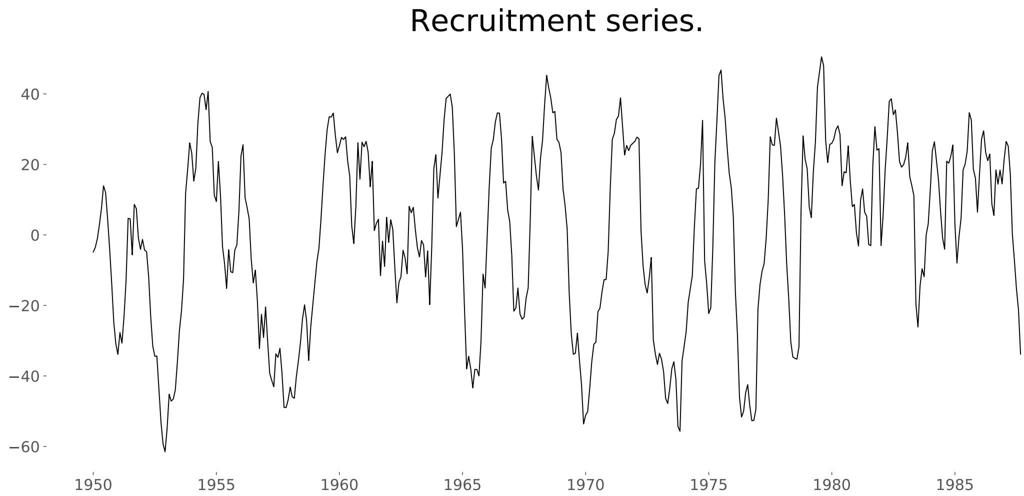
Figur 1: Tidsrekken plottet med vektet periodogram. Her kan man se forskjellige periodisiteter rundt f = 0.5/år og en på f = 1/år. Frekvensaksen har blitt ganget med [12 måneder/år] for å få enhet 1/år.

1.2 b)

Her trekker vi fra midlere sesongvariasjoner for å gjøre tidsrekken stasjonær. Resultatet er plottet i figur(2). Dette ble gjort ved å trekke fra gjennomsnittet i hver måned fra alle månede igjennom hele datasettet.

1.3 c-d)

Vi har plott av ACF og PACF i figur(3). Her kan man observere at ACF-plottet har den karakteristiske AR() modellen der det konvergerer mot null når $h \to 0$, men denne gir ingen indikasjon

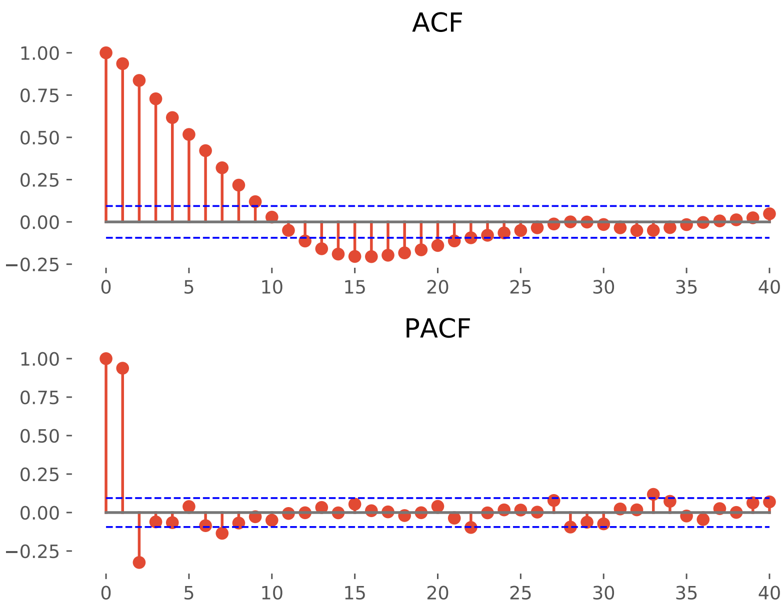


Figur 2: Trukket fra midlere sesongvariasjoner slik at tidsrekken blir stasjonær.

på orden av AR. I PACF ser man at det er en korrelasjon ved $h = 1$ og $h = 2$, resten ser ut til å være hvit støy da dette ligger under 95% konfidensinterval gitt i ligning(1.2). På grunn av den klare indikasjonen på korrelasjon ved $h = 1$ og $h = 2$ kan vi si at vi har en AR(2) eller ARMA(2, 0) prosess.

$$\sigma_w = \frac{2}{\sqrt{N}} \tag{1.2}$$

N er lengden på tidsserien.[p. 31 Shumway [2017]]



Figur 3: Plot av ACF og PACF for den stasjonære tidsrekken i figur(2)

1.4 e)

Ved bruk av statsmodels-pakken i python kan vi simulere en ARMA-modell med gitte parametere, i figur(4) ser vi en resultatet fra en ARMA(2,0) modell. Fra utskriften fikk vi modellen gitt i ligning(1.3).

$$\hat{x}_n = \hat{\alpha} + \phi_1 \hat{x}_{n-1} + \phi_2 \hat{x}_{n-2} \tag{1.3}$$

der $\hat{\alpha}_{(4.777)} = -0.7019$, $\phi_{1(0.044)} = 1.2483$ og $\phi_{2(0.044)} = -0.3313$. Verdiene gitt i parantes er standardfeilen i parameterene. Ser man på alpha (konstant assosiert med forventningen) kan denne feilen føre til at vi egentlig kan ha en forventing som er positiv. Her har det blitt brukt en *hatt* på tidsrekken for å vise at det er et estimat. Fra utskriften i figur(4) har vi at $\sigma_w = 8.653$, dette er den estimerte variansen på den hvite støyen slik man har fordelingen $N \sim (0, \sigma_w^2)$. Dette kan brukes til å finne *mean-square prediction error* som er gitt ved ligning(1.4),

$$P_{n+m}^n = E\Big[\sigma_w^2 \sum_{j=0}^{m-1} \psi_j^2\Big] \tag{1.4}$$

der ψ er vektene gitt fra modellen slik at $\phi(z)\psi(z) = \theta(z)$. Her er $\psi(z)$ og $\theta(z)$ de karakteristiske ligningene til AR og MA modellen og $\psi(z)$ er vektene gitt ved $\psi(z) = (1 + \psi_1 z + \psi_2 z^2 + \cdots + \psi_j z^j + \cdots)$

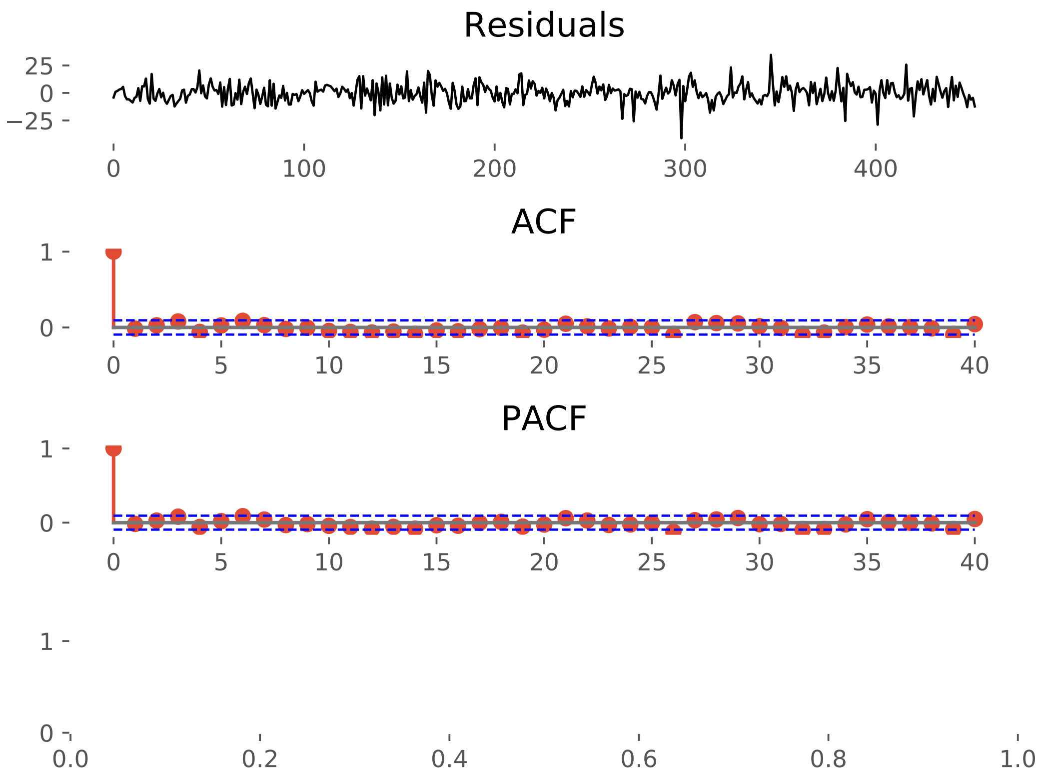
```
                              ARMA Model Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  453
Model:                     ARMA(2, 0)   Log Likelihood               -1616.790
Method:                       css-mle   S.D. of innovations              8.564
Date:                Tue, 16 Apr 2019   AIC                           3241.579
Time:                        12:27:46   BIC                           3258.043
Sample:                             0   HQIC                          3248.066

==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.7019      4.773     -0.147      0.883     -10.057       8.653
ar.L1.y        1.2483      0.044     28.159      0.000       1.161       1.335
ar.L2.y       -0.3313      0.044     -7.471      0.000      -0.418      -0.244
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1            1.1555           +0.0000j            1.1555            0.0000
AR.2            2.6120           +0.0000j            2.6120            0.0000
-----------------------------------------------------------------------------
```

Figur 4: Utskrift av ARMA-model resultat gitt fra statsmodels-pakken. Merk her at det har blitt brukt css-MLE for å estimere parametere.(conditional sum of squares - most ikelihood estimator.) `https://www.statsmodels.org/dev/generated/statsmodels.tsa.arima_model.ARMA.html`

2 Oppgave 2

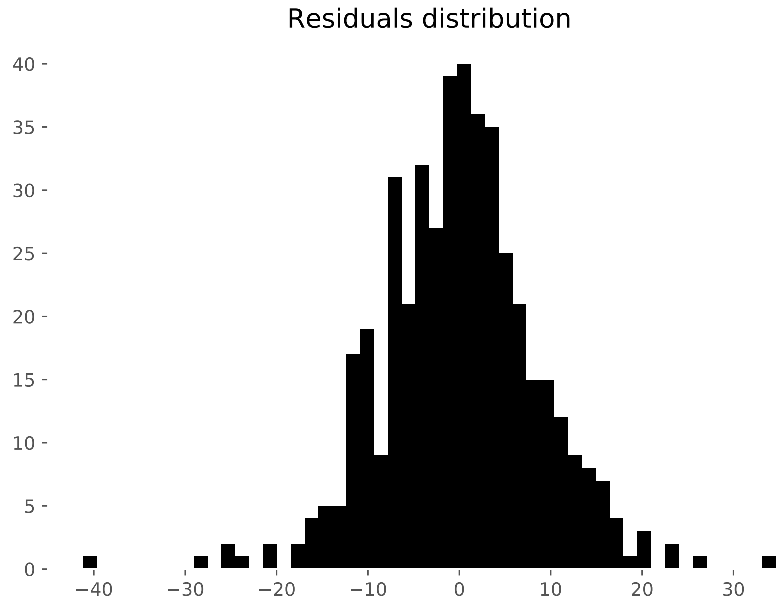
2.1 a-b)

Ved å trekke ifra modellen(uten det hvite støyleddet) på den orginale tidsserien får vi residualene/feilene, resultatet er vist i figur(5). Fra plottet av ACF og PACF ser man at alt ligger under 95% konfidensintervallet for hvit støy, dette kan være en indikator på at modellen er god fordi vi kun står igjen med hvit støy. MERK: *Her har vi kun brukt ACF og PACF opp til lag $h = 40$ det kan være at det ligger noe utafor dette(for eksempel at vi har korrelasjon mellom lag x_t, x_{t+100}), dette gjelder også for det ACF og PACF i de tidligere oppgavene.*



Figur 5: Plot av ACF og PACF av residualene. Her er det ingen korrelasjon og kan dermed anta hvit støy.

I figur(6) er det plottet et histogram av resiudalene. Histogrammet kan avsløre fordelingen til dataen, som i dette tilfelle viser at det nesten er normalfordelt.

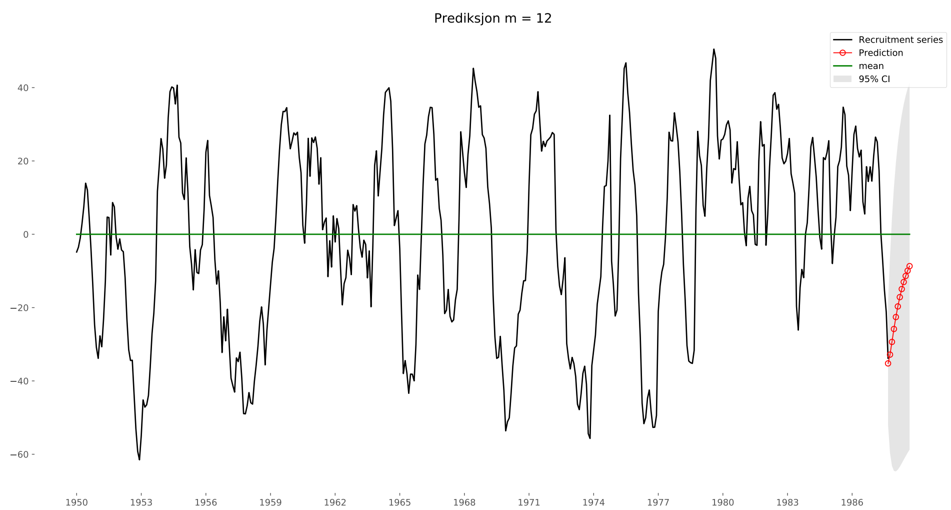


Figur 6: Ved bruk av histogram kan man se fordelingen til dataen. Her kan man se at residualene ligner på en normalfordeling.

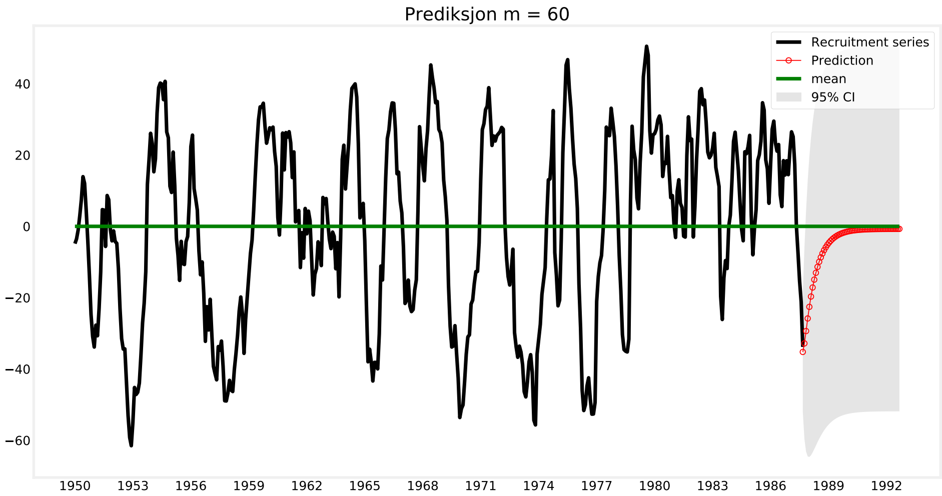
3 Oppgave 3

3.1 a-b)

Prediksjon har blitt gjort med statsmodels sin ARMA-predict funksjon i python. Resultatet med 12 steg prediksjon er vist i figure(7). Feilen konvergerer veldig fort til variansen av tidsrekken σ_x som vist i figur(8). Man kan også observere at prediksjonen konvergerer mot gjennomsnittet av tidsrekken.



Figur 7: Prediksjon med M = 12 måneder.



Figur 8: Prediksjon med M = 60 måneder. Her kan man se at feilen konvergerer mot variansen til tidsserien når man bruker høy *m*, og prediksjonen går mot gjennomsnittet (grønn linje).

4 Appendix

```python
from statsmodels.tsa.stattools import acf, pacf, ccf
from statsmodels.tsa.arima_process import arma2ma, arma2ar
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import os
import statsmodels as sm
plt.style.use('fivethirtyeight')
plt.rcParams['axes.facecolor']='white'
plt.rcParams['savefig.facecolor']='white'
plt.rcParams['axes.grid']='off'

# Load data
rec = pd.read_csv('data/rec.txt', delimiter='\t')
rec_df = pd.DataFrame(rec)
time = np.copy(rec_df['year'])
X = np.copy(rec_df['recruitment'])
```

Figur 9: Load files

```python
    def w_periodogram(x, dt = 1):
    """Windowed periodogram"""
    #x = np.pad(x, (0,300), 'constant')
    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[int(N/2):], spectrum[int(N/2):]

# Task A

# Find periodogram
freq, periodogram_X = w_periodogram(X)


# Plot data
fig, ax = plt.subplots(2,1)
ax[0].plot(time, X, color = 'black')
ax[0].set_title('Recruitment series')
ax[0].set_xlabel('Year')
ax[0].set_ylabel('Recruitment')
ax[1].plot(freq[2:]*12, periodogram_X[2:], color = 'black')
ax[1].set_title('Periodogram(hann windowed)')
ax[1].set_xlabel('Frequency $\Delta f =$ year')
ax[1].set_ylabel('Power')
ax[1].set_xticks([x for x in np.arange(0, 6.5, 1/2)])
plt.tight_layout()
plt.savefig('rapport/task_a.pdf')
plt.show()

# Task B

def remove_season(x):
    C = np.zeros(12)
    for m in range(0,12):
        C[m] = np.mean(x[m::12])

    # repeat C to create a periodic signal of equal length or longer than the dataset
    repC = np.tile(C, int(np.ceil(len(x)/12)))
    # compute residual (by subtracting periodic signal)
    X = x - repC[:len(x)]
    return X

# Make stationary
X_remseason = remove_season(X)

plt.plot(time, X_remseason, color = 'black')
plt.title('Recruitment series.')
plt.tight_layout()
plt.savefig('rapport/task_b.pdf')
plt.show()

# TASK C

# make whitenoise Confidens intervall
wt_line = 2*np.tile(1/np.sqrt(len(X_remseason)), 41)


# Plot
fig, ax = plt.subplots(2,1)
# ax[0].plot(time, X_remseason)
# ax[0].set_title('Stasjonre tidsserien')
ax[0].stem(acf(X_remseason))
ax[0].set_title('ACF')
ax[0].plot(wt_line, '--', color = 'red', linewidth = 1); ax[0].plot(-wt_line, '--', color = 'red', linewidth = 1)
ax[1].stem(pacf(X_remseason))
ax[1].set_title('PACF')
ax[1].plot(wt_line, '--', color = 'red', linewidth = 1); ax[1].plot(-wt_line, '--', color = 'red', linewidth = 1)
plt.tight_layout()
```

```
        # Oppgave 2


        # Task A

        # Create model.
        model = sm.tsa.arima_model.ARIMA(X_remseason, order=(2, 0, 0))
        model_fit = model.fit()

        # Get residuals
        res = model_fit.resid

        # Plot
        fig, ax = plt.subplots(3,1)
        ax[0].plot(res, color = 'black')
        ax[0].set_title('Residuals')
        ax[1].stem(acf(res))
        ax[1].plot(wt_line, '--', color = 'red', linewidth = 1); ax[1].plot(-wt_line, '--', color = 'red', linewidth = 1)
        ax[1].set_title('ACF')
        ax[2].stem(pacf(res))
        ax[2].plot(wt_line, '--', color = 'red', linewidth = 1); ax[2].plot(-wt_line, '--', color = 'red', linewidth = 1)
        ax[2].set_title('PACF')
        plt.tight_layout()
        plt.savefig('rapport/task_2a.pdf')
        plt.show()

```

Figur 11: Task 2

```
    # Set time parameters
    year = 1
    M = 12*year # 12*2 months (2 years)

    # Get forecast
    forecast, stderr, conf_int = model_fit.forecast(steps = M)
    # gir ut forecast, std, (1-alpha)% konfidensintervall. Default: 95% konfidensintervall


    # Get time arrays
    sliced_time = time
    sliced_X = X_remseason
    time_forecast = np.linspace(sliced_time[-1], sliced_time[-1] + M/12, M)
    tot_time = np.linspace(sliced_time[0], time_forecast[-1], len(X_remseason)+len(forecast))

    # Plot
    plt.figure(figsize = [15,8])
    plt.plot(sliced_time, sliced_X, color = 'black', label = 'Recruitment series')
    plt.plot(time_forecast, forecast, '-o', mfc='none', color = 'red', linewidth = '1', label = 'Prediction' )
    plt.plot(tot_time, np.tile(np.mean(X_remseason), reps = len(X_remseason)+len(forecast)), color = 'green', label = 'mean')
    plt.fill_between(time_forecast, conf_int[:,0], conf_int[:,1], facecolor = (0.5, 0.5, 0.5, 0.2), label = '95% CI')
    plt.xticks([x for x in np.arange(sliced_time[0], sliced_time[-1]+ M/12, 3)])
    plt.legend(loc = 'best')
    plt.title('Prediksjon m = %s'%M)
    plt.tight_layout()
    plt.savefig('rapport/task_33.pdf')
    #plt.show()
    plt.close()

```

Figur 12: Task 3

5 Referanser

Robert H. Shumway. *Time Series Analysis and its Applications*. Springer texts in statistics, fourth edition, 2017.