Granger causality

In [1]:

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

import warnings
warnings.filterwarnings('ignore')

import re
```

In [2]:

```
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.seasonal import seasonal_decompose
from pmdarima import auto_arima

from statsmodels.stats.diagnostic import acorr_ljungbox
from statsmodels.tsa.stattools import adfuller, kpss, grangercausalitytests

from sklearn.metrics import mean_absolute_percentage_error


def difference(dataset, interval=1):
    diff = list()
    for i in range(interval, len(dataset)):
        value = dataset[i] - dataset[i - interval]
        diff.append(value)
    return pd.Series(diff)
```

In [3]:

```
def grangers_causation_matrix(data, variables, verbose=False, maxlag = 12):
    """Check Granger Causality matrix.
    The rows are restponse and columns are predictors.
    P-Values < 0.05, implies that the X does not cause Y to be rejected.

    data      : dataframe containing the time series variables
    variables : columns.
    """
    df = pd.DataFrame(np.zeros((len(variables), len(variables))), columns=variables, index=variables)
    for c in df.columns:
        for r in df.index:
            test_result = grangercausalitytests(data[[r, c]], maxlag=maxlag, verbose=verbose)
            p_values = [round(test_result[i+1][0]['ssr_chi2test'][1],4) for i in range(maxlag)]
            if not verbose:
                print(f'Y = {r}, X = {c}, P Values = {p_values}')
            min_p_value = np.min(p_values)
            df.loc[r, c] = min_p_value
    df.columns = [var + '_x' for var in variables]
    df.index = [var + '_y' for var in variables]
    return df
```

In [ ]:

```
# we tried to descibe the dynamics of time series at different points but there can be different factors that are not directly known
# we want to include predictor variables
# Granger causality - whether one time-series is usefull in predicting another one
```

In [4]:

```
dfSamples = pd.read_csv("../Lab8/Wage and Inflation data/Mehra.csv", index_col = 0, parse
_dates=True)
dfSamples.index.freq = 'QS'
dfSamples
```

Out[4]:

	rgnp	pgnp	ulc	gdfco	gdf	gdfim	gdfcf	gdfce
date								
1959-01-01	1606.4	1608.3	47.5	36.9	37.4	26.9	32.3	23.1
1959-04-01	1637.0	1622.2	47.5	37.4	37.5	27.0	32.2	23.4
1959-07-01	1629.5	1636.2	48.7	37.6	37.6	27.1	32.4	23.4
1959-10-01	1643.4	1650.3	48.8	37.7	37.8	27.1	32.5	23.8
1960-01-01	1671.6	1664.6	49.1	37.8	37.8	27.2	32.4	23.8
...	...	...	...	...	...	...	...	...
1988-07-01	4042.7	3971.9	179.6	131.5	124.9	106.2	123.5	92.8
1988-10-01	4069.4	3995.8	181.3	133.3	126.2	107.3	124.9	92.9
1989-01-01	4106.8	4019.9	184.1	134.8	127.7	109.5	126.6	94.0
1989-04-01	4132.5	4044.1	186.1	134.8	129.3	111.1	129.0	100.6
1989-07-01	4162.9	4068.4	187.4	137.2	130.2	109.8	129.9	98.2

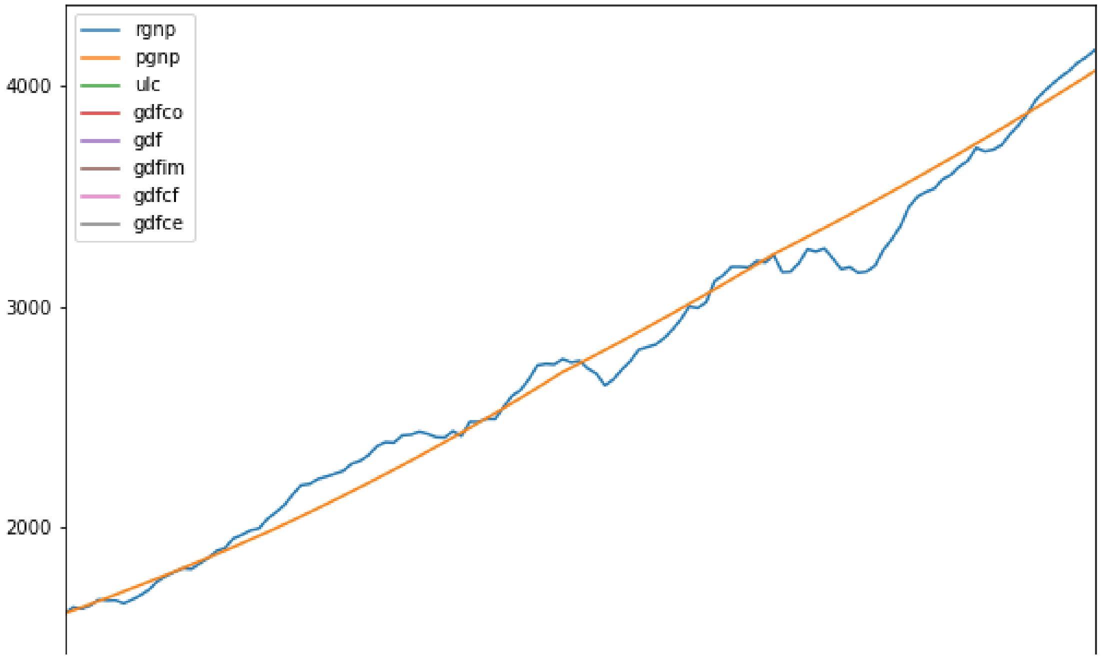
123 rows × 8 columns

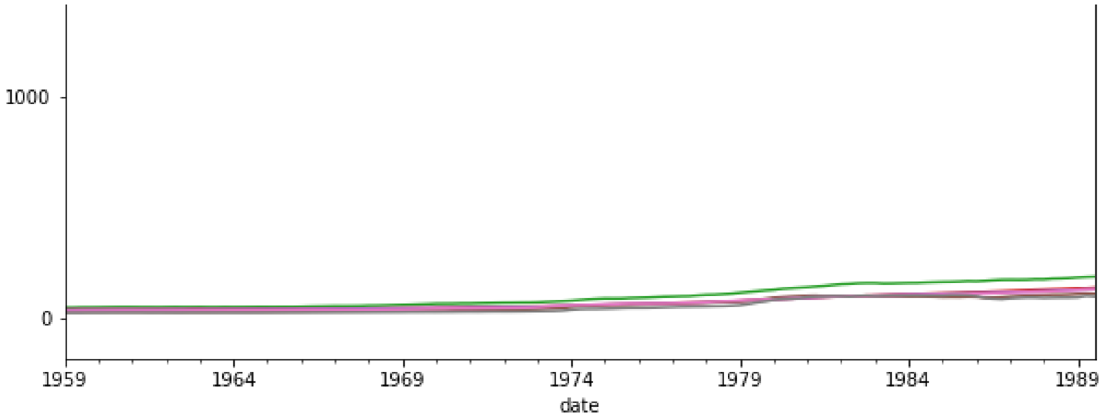
In [5]:

```
fig, ax = plt.subplots(1, figsize=(10,10))
dfSamples['rgnp'].plot(legend = True)
dfSamples['pgnp'].plot(ax=ax, legend = True)
dfSamples['ulc'].plot(ax=ax, legend = True)
dfSamples['gdfco'].plot(ax=ax, legend = True)
dfSamples['gdf'].plot(ax=ax, legend = True)
dfSamples['gdfim'].plot(ax=ax, legend = True)
dfSamples['gdfcf'].plot(ax=ax, legend = True)
dfSamples['gdfce'].plot(ax=ax, legend = True)
```

Out[5]:

```
<AxesSubplot:xlabel='date'>
```



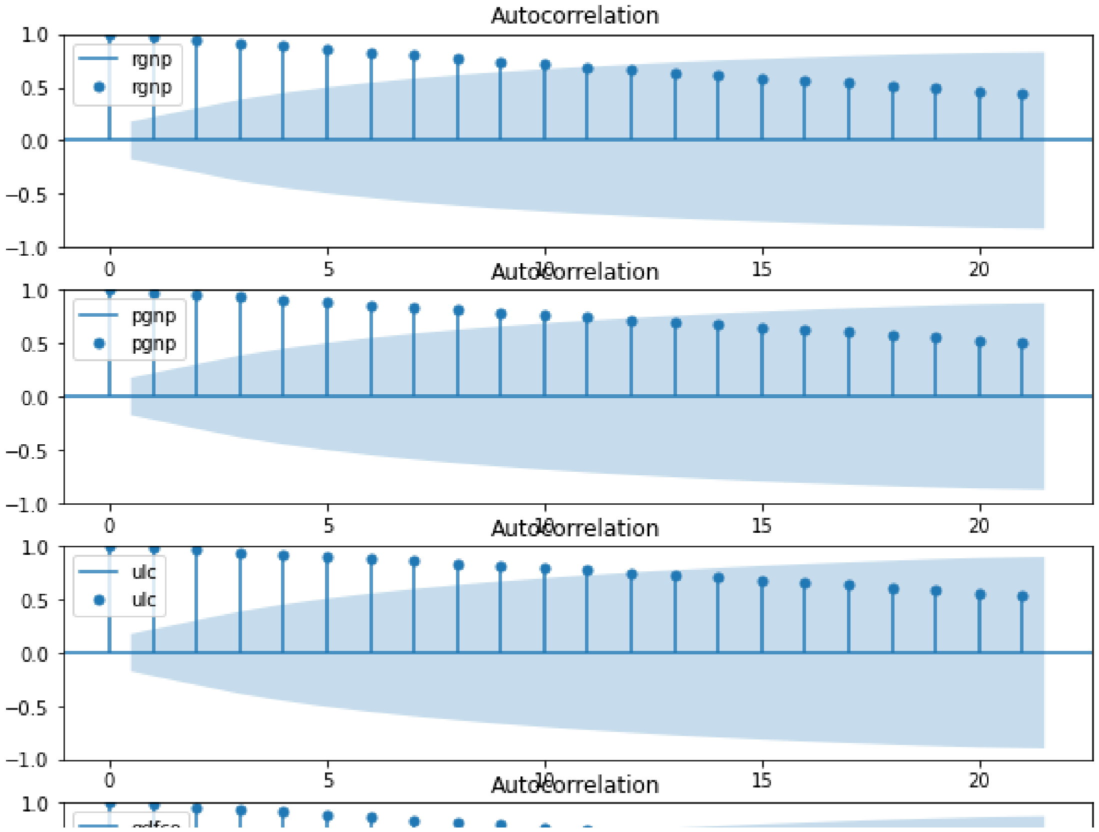


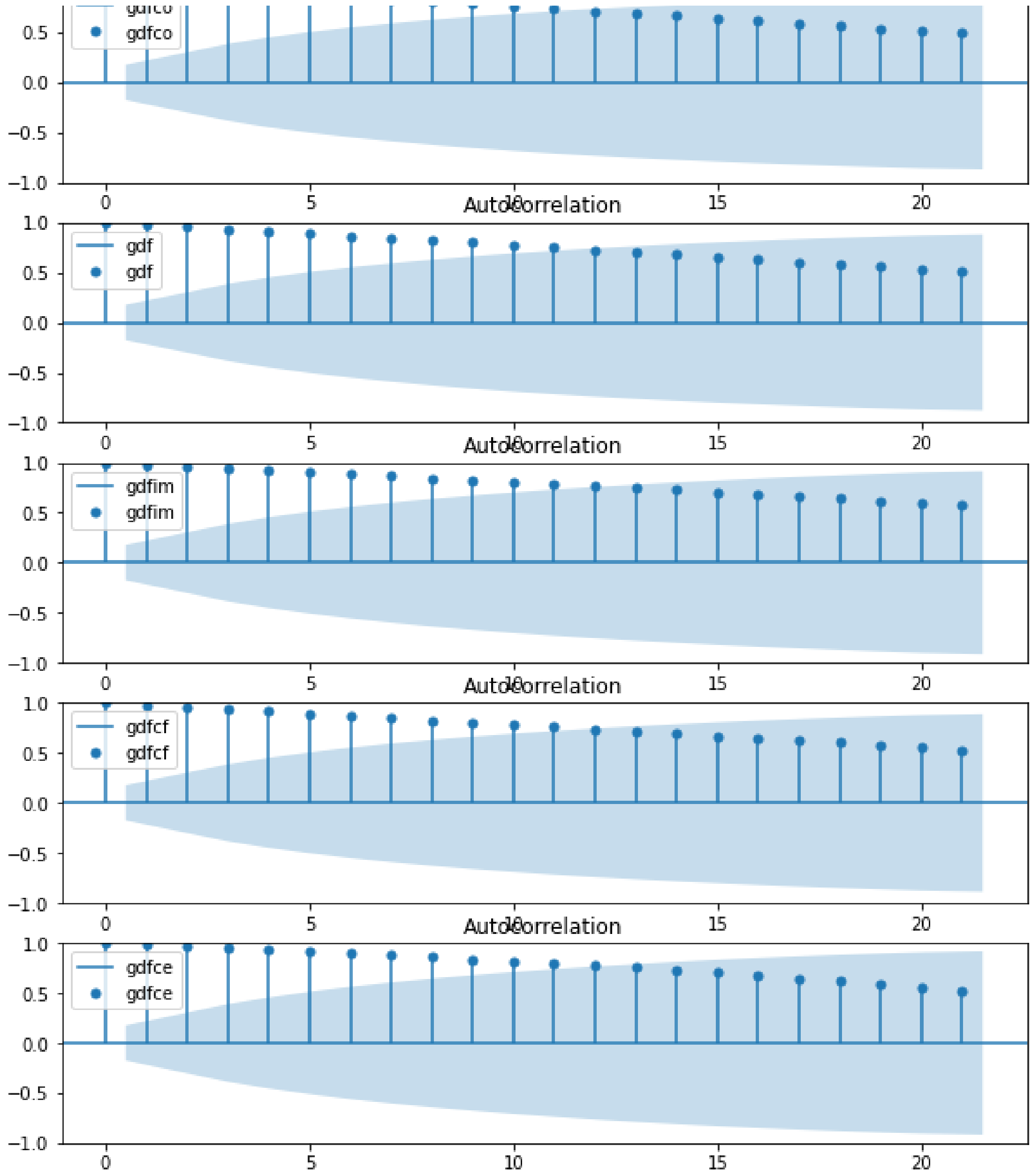
Time series differencing to obtain stationary statistics and to make Granger causaliy test. In order to see that we check the Augmented Dickey–Fuller test again.

In [6]:

```
fig, ax = plt.subplots(len(dfSamples.columns), figsize = (10,20))
for i,col in enumerate(dfSamples.columns):
    print("test: ", col)
    print(f"p_value = {adfuller(dfSamples[col])[1]}")
    plot_acf(dfSamples[col], ax = ax[i], label = col)
    ax[i].legend()
```

```
test:  rgnp
p_value = 0.9886037114305949
test:  pgnp
p_value = 0.9964612064726062
test:  ulc
p_value = 0.997099172882968
test:  gdfco
p_value = 0.9870138472785789
test:  gdf
p_value = 0.9953249001658118
test:  gdfim
p_value = 0.9386740229058363
test:  gdfcf
p_value = 0.998051164027281
test:  gdfce
p_value = 0.8144015307501952
```





In [8]:

```
dfSamples2 = dfSamples.copy()
orders = []
for i,col in enumerate(dfSamples.columns):
    print("test: ", col)
    model = auto_arima(dfSamples[col], seasonal = False, m = 1, trace = True, verbose=Fa
lse) #.summary()
    print("\n\n-----------------------------------------------------------------\n\n")
    order = model.get_params()['order']
    d=order[1]
    print(f"differencing order is d={d}")
    orders.append(d)
```

```
test:  rgnp
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=1149.375, Time=0.29 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=1156.412, Time=0.02 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=1147.271, Time=0.10 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=1150.510, Time=0.10 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=1211.180, Time=0.02 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=1145.911, Time=0.10 sec
 ARIMA(3,1,0)(0,0,0)[0] intercept   : AIC=1147.563, Time=0.12 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=1147.412, Time=0.26 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=1147.085, Time=0.15 sec
 ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=1149.561, Time=0.17 sec
 ARIMA(2,1,0)(0,0,0)[0]             : AIC=1158.542, Time=0.05 sec

Best model:  ARIMA(2,1,0)(0,0,0)[0] intercept
Total fit time: 1.375 seconds
```

```
-----------------------------------------------------------------


differencing order is d=1
test:  pgnp
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.48 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=207.958, Time=0.04 sec
 ARIMA(1,2,0)(0,0,0)[0] intercept   : AIC=209.863, Time=0.03 sec
 ARIMA(0,2,1)(0,0,0)[0] intercept   : AIC=209.862, Time=0.06 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=208.755, Time=0.02 sec
 ARIMA(1,2,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.36 sec

Best model:  ARIMA(0,2,0)(0,0,0)[0] intercept
Total fit time: 0.990 seconds


-----------------------------------------------------------------


differencing order is d=2
test:  ulc
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.46 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=349.442, Time=0.02 sec
 ARIMA(1,2,0)(0,0,0)[0] intercept   : AIC=331.963, Time=0.05 sec
 ARIMA(0,2,1)(0,0,0)[0] intercept   : AIC=321.226, Time=0.07 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=347.455, Time=0.02 sec
 ARIMA(1,2,1)(0,0,0)[0] intercept   : AIC=321.319, Time=0.10 sec
 ARIMA(0,2,2)(0,0,0)[0] intercept   : AIC=321.811, Time=0.10 sec
 ARIMA(1,2,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.36 sec
 ARIMA(0,2,1)(0,0,0)[0]             : AIC=319.346, Time=0.03 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=319.551, Time=0.05 sec
 ARIMA(0,2,2)(0,0,0)[0]             : AIC=319.991, Time=0.05 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=329.988, Time=0.02 sec
 ARIMA(1,2,2)(0,0,0)[0]             : AIC=319.733, Time=0.16 sec

Best model:  ARIMA(0,2,1)(0,0,0)[0]
Total fit time: 1.495 seconds


-----------------------------------------------------------------


differencing order is d=2
test:  gdfco
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0] intercept   : AIC=83.921, Time=0.28 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=139.448, Time=0.04 sec
 ARIMA(1,2,0)(0,0,0)[0] intercept   : AIC=105.962, Time=0.06 sec
 ARIMA(0,2,1)(0,0,0)[0] intercept   : AIC=84.788, Time=0.12 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=137.614, Time=0.02 sec
 ARIMA(1,2,2)(0,0,0)[0] intercept   : AIC=87.576, Time=0.25 sec
 ARIMA(2,2,1)(0,0,0)[0] intercept   : AIC=84.541, Time=0.21 sec
 ARIMA(3,2,2)(0,0,0)[0] intercept   : AIC=85.920, Time=0.44 sec
 ARIMA(2,2,3)(0,0,0)[0] intercept   : AIC=85.890, Time=0.37 sec
 ARIMA(1,2,1)(0,0,0)[0] intercept   : AIC=84.933, Time=0.11 sec
 ARIMA(1,2,3)(0,0,0)[0] intercept   : AIC=84.105, Time=0.22 sec
 ARIMA(3,2,1)(0,0,0)[0] intercept   : AIC=86.365, Time=0.24 sec
 ARIMA(3,2,3)(0,0,0)[0] intercept   : AIC=87.837, Time=0.58 sec
 ARIMA(2,2,2)(0,0,0)[0]             : AIC=82.863, Time=0.18 sec
 ARIMA(1,2,2)(0,0,0)[0]             : AIC=87.263, Time=0.16 sec
 ARIMA(2,2,1)(0,0,0)[0]             : AIC=83.394, Time=0.09 sec
 ARIMA(3,2,2)(0,0,0)[0]             : AIC=84.849, Time=0.28 sec
 ARIMA(2,2,3)(0,0,0)[0]             : AIC=84.690, Time=0.20 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=84.062, Time=0.06 sec
 ARIMA(1,2,3)(0,0,0)[0]             : AIC=82.808, Time=0.18 sec
 ARIMA(0,2,3)(0,0,0)[0]             : AIC=83.871, Time=0.12 sec
 ARIMA(1,2,4)(0,0,0)[0]             : AIC=84.718, Time=0.19 sec
 ARIMA(0,2,2)(0,0,0)[0]             : AIC=82.996, Time=0.07 sec
 ARIMA(0,2,4)(0,0,0)[0]             : AIC=83.736, Time=0.10 sec
 ARIMA(2,2,4)(0,0,0)[0]             : AIC=inf, Time=0.61 sec

Best model:  ARIMA(1,2,3)(0,0,0)[0]
Total fit time: 5.188 seconds


-----------------------------------------------------------------


differencing order is d=2
test:  gdf
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0] intercept   : AIC=-30.678, Time=0.24 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=-21.577, Time=0.03 sec
 ARIMA(1,2,0)(0,0,0)[0] intercept   : AIC=-30.132, Time=0.06 sec
 ARIMA(0,2,1)(0,0,0)[0] intercept   : AIC=-33.420, Time=0.09 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=-23.465, Time=0.03 sec
 ARIMA(1,2,1)(0,0,0)[0] intercept   : AIC=-31.420, Time=0.11 sec
 ARIMA(0,2,2)(0,0,0)[0] intercept   : AIC=-31.421, Time=0.12 sec
 ARIMA(1,2,2)(0,0,0)[0] intercept   : AIC=-29.809, Time=0.25 sec
 ARIMA(0,2,1)(0,0,0)[0]             : AIC=-34.916, Time=0.04 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=-32.918, Time=0.07 sec
 ARIMA(0,2,2)(0,0,0)[0]             : AIC=-32.920, Time=0.06 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=-31.834, Time=0.04 sec
 ARIMA(1,2,2)(0,0,0)[0]             : AIC=-31.286, Time=0.12 sec

Best model:  ARIMA(0,2,1)(0,0,0)[0]
Total fit time: 1.281 seconds


-----------------------------------------------------------------


differencing order is d=2
test:  gdfim
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.43 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=409.138, Time=0.02 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=322.229, Time=0.04 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=348.274, Time=0.05 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=437.719, Time=0.02 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=324.177, Time=0.08 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=324.156, Time=0.09 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.39 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=323.985, Time=0.02 sec

Best model:  ARIMA(1,1,0)(0,0,0)[0] intercept
Total fit time: 1.142 seconds


-----------------------------------------------------------------


differencing order is d=1
test:  gdfcf
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0] intercept   : AIC=219.397, Time=0.20 sec
 ARIMA(0,2,0)(0,0,0)[0] intercept   : AIC=254.358, Time=0.03 sec
 ARIMA(1,2,0)(0,0,0)[0] intercept   : AIC=238.850, Time=0.04 sec
 ARIMA(0,2,1)(0,0,0)[0] intercept   : AIC=224.371, Time=0.07 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=252.376, Time=0.02 sec
 ARIMA(1,2,2)(0,0,0)[0] intercept   : AIC=219.222, Time=0.18 sec
 ARIMA(0,2,2)(0,0,0)[0] intercept   : AIC=223.989, Time=0.11 sec
 ARIMA(1,2,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.26 sec
 ARIMA(1,2,3)(0,0,0)[0] intercept   : AIC=220.978, Time=0.22 sec
 ARIMA(0,2,3)(0,0,0)[0] intercept   : AIC=223.698, Time=0.14 sec
 ARIMA(2,2,1)(0,0,0)[0] intercept   : AIC=218.379, Time=0.12 sec
 ARIMA(2,2,0)(0,0,0)[0] intercept   : AIC=220.590, Time=0.07 sec
 ARIMA(3,2,1)(0,0,0)[0] intercept   : AIC=219.781, Time=0.24 sec
 ARIMA(3,2,0)(0,0,0)[0] intercept   : AIC=220.728, Time=0.09 sec
 ARIMA(3,2,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.60 sec
 ARIMA(2,2,1)(0,0,0)[0]             : AIC=216.510, Time=0.10 sec
 ARIMA(1,2,1)(0,0,0)[0]             : AIC=223.029, Time=0.09 sec
 ARIMA(2,2,0)(0,0,0)[0]             : AIC=218.780, Time=0.05 sec
 ARIMA(3,2,1)(0,0,0)[0]             : AIC=217.930, Time=0.14 sec
 ARIMA(2,2,2)(0,0,0)[0]             : AIC=217.582, Time=0.10 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=236.919, Time=0.03 sec
 ARIMA(1,2,2)(0,0,0)[0]             : AIC=217.693, Time=0.08 sec
 ARIMA(3,2,0)(0,0,0)[0]             : AIC=218.869, Time=0.05 sec
 ARIMA(3,2,2)(0,0,0)[0]             : AIC=212.220, Time=0.23 sec
 ARIMA(4,2,2)(0,0,0)[0]             : AIC=214.027, Time=0.33 sec
 ARIMA(3,2,3)(0,0,0)[0]             : AIC=214.108, Time=0.28 sec
 ARIMA(2,2,3)(0,0,0)[0]             : AIC=217.704, Time=0.22 sec
 ARIMA(4,2,1)(0,0,0)[0]             : AIC=212.679, Time=0.21 sec
 ARIMA(4,2,3)(0,0,0)[0]             : AIC=215.746, Time=0.58 sec

Best model:  ARIMA(3,2,2)(0,0,0)[0]
Total fit time: 4.888 seconds


-----------------------------------------------------------------



differencing order is d=2
test:  gdfce
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=479.630, Time=0.22 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=511.497, Time=0.01 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=478.132, Time=0.28 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=484.108, Time=0.06 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=521.239, Time=0.02 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=479.902, Time=0.06 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=479.784, Time=0.08 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=480.951, Time=0.27 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=479.483, Time=0.02 sec

Best model:  ARIMA(1,1,0)(0,0,0)[0] intercept
Total fit time: 1.017 seconds


-----------------------------------------------------------------



differencing order is d=1
```

In [9]:

```
orders[-1] = 2
maxOrder = np.max(orders)
ordersDic = {dfSamples.columns[i]:orders[i] for i in range(len(orders))}
print(f'max order is {maxOrder}')
dfSamples2 = dfSamples.copy()
for i,col in enumerate(dfSamples.columns):
    tmp = dfSamples[col]
    ord = int(orders[i])
    for j in range(ord):
        tmp = difference(tmp)
    tmp = list(tmp)
    for j in range(ord):
        tmp = [np.nan] + tmp
    dfSamples2[col] = np.array(tmp)

dfSamples2 = dfSamples2.dropna()
dfSamples2
```

```
max order is 2
```

Out[9]:

	rgnp	pgnp	ulc	gdfco	gdf	gdfim	gdfcf	gdfce
date								
1959-07-01	-7.5	0.1	1.2	-3.000000e-01	0.0	0.1	0.3	-0.3

	rgnp	pgnp	ulc	gdfco	gdf	gdfim	gdfcf	gdfce
date								
1959-07-01	-7.5	0.1	1.2	-3.000000e-01	0.0	0.1	0.3	-0.3
1959-10-01	13.9	0.1	-1.1	-1.000000e-01	0.1	0.0	-0.1	0.4
1960-01-01	28.2	0.2	0.2	-7.105427e-15	-0.2	0.1	-0.2	-0.4
1960-04-01	-4.8	0.1	0.2	1.000000e-01	0.2	0.2	0.5	0.1
1960-07-01	1.6	0.1	-0.1	-1.000000e-01	-0.1	0.0	-0.3	0.1
...	...	...	...	...	...	...	...	...
1988-07-01	32.0	0.2	-2.1	-4.000000e-01	0.2	0.1	0.9	-0.3
1988-10-01	26.7	0.1	0.8	6.000000e-01	-0.3	1.1	-1.1	-0.4
1989-01-01	37.4	0.2	1.1	-3.000000e-01	0.2	2.2	0.3	1.0
1989-04-01	25.7	0.1	-0.8	-1.500000e+00	0.1	1.6	0.7	5.5
1989-07-01	30.4	0.1	-0.7	2.400000e+00	-0.7	-1.3	-1.5	-9.0

121 rows × 8 columns

In [10]:

```
ordersDic
# order of gdfce shall be 2 or more
```

Out[10]:

```
{'rgnp': 1,
 'pgnp': 2,
 'ulc': 2,
 'gdfco': 2,
 'gdf': 2,
 'gdfim': 1,
 'gdfcf': 2,
 'gdfce': 2}
```

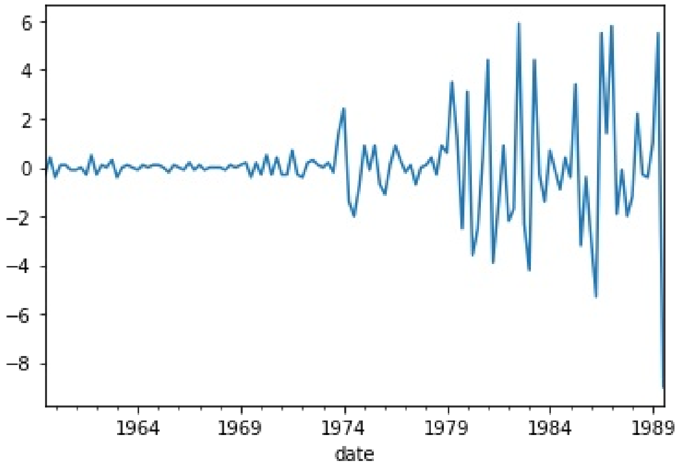
Check the test again for stationarity

In [11]:

```
dfSamples2['gdfce'].plot()
```

Out[11]:

```
<AxesSubplot:xlabel='date'>
```

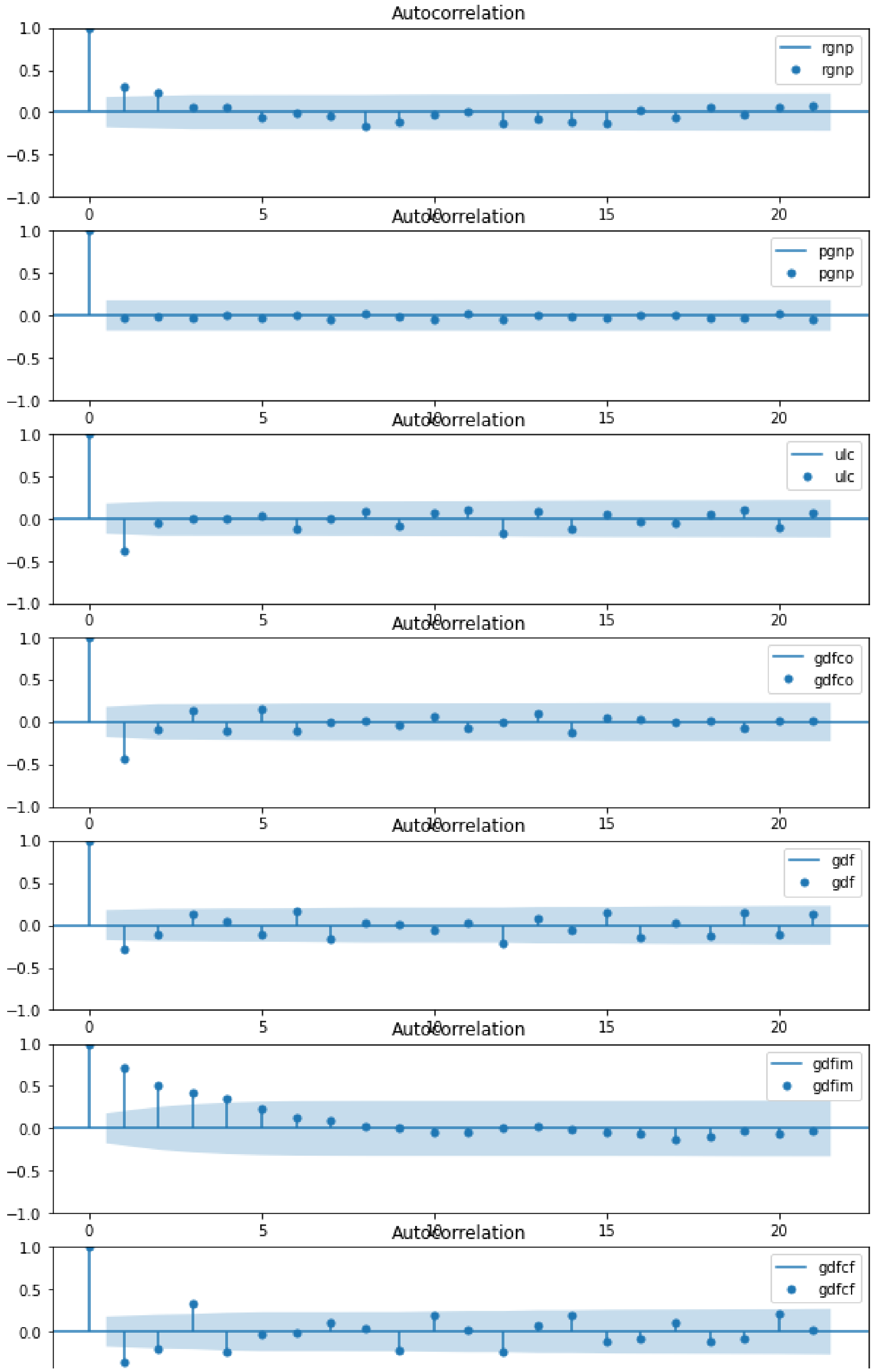


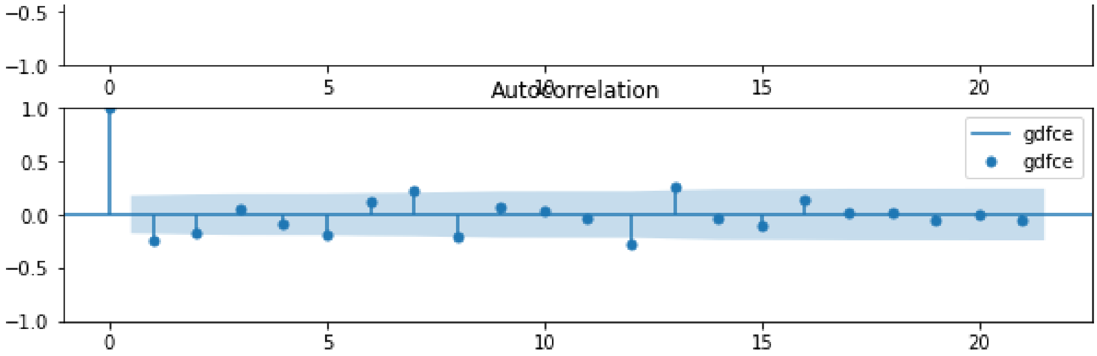
In [12]:

```
fig, ax = plt.subplots(len(dfSamples2.columns), figsize = (10,20))
for i,col in enumerate(dfSamples2.columns):
    print("test: ", col)
    print(f"p_value = {adfuller(dfSamples2[col])[1]}")
    plot_acf(dfSamples2[col], ax = ax[i], label = col)
    ax[i].legend()
```

```
test:  rgnp
```

```
p_value = 3.3385082936364718e-06
test:  pgnp
p_value = 2.6204782825958454e-20
test:  ulc
p_value = 1.698194036505408e-14
test:  gdfco
p_value = 2.4994349960269606e-14
test:  gdf
p_value = 0.00039702611077686476
test:  gdfim
p_value = 0.0005850011668710128
test:  gdfcf
p_value = 2.1400722424590324e-10
test:  gdfce
p_value = 0.00032807186699190094
```





We would like to check whether the information between different datasets can help predict the other ones

In [13]:

```
granger = grangers_causation_matrix(dfSamples2, dfSamples2.columns, False)
granger
```

```
Y = rgnp, X = rgnp, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.
0]
Y = pgnp, X = rgnp, P Values = [0.7618, 0.1511, 0.2545, 0.3159, 0.4582, 0.5171, 0.5985, 0
.0676, 0.051, 0.0504, 0.0461, 0.007]
Y = ulc, X = rgnp, P Values = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
]
Y = gdfco, X = rgnp, P Values = [0.9136, 0.4359, 0.7276, 0.3431, 0.2975, 0.328, 0.0458, 0
.0603, 0.1001, 0.0361, 0.0161, 0.0079]
Y = gdf, X = rgnp, P Values = [0.2351, 0.2986, 0.181, 0.1191, 0.1928, 0.1597, 0.2463, 0.3
125, 0.4089, 0.0308, 0.0348, 0.1099]
Y = gdfim, X = rgnp, P Values = [0.4785, 0.6606, 0.8615, 0.0861, 0.0712, 0.1403, 0.0828,
0.1323, 0.1811, 0.2483, 0.3231, 0.18]
Y = gdfcf, X = rgnp, P Values = [0.7522, 0.7968, 0.4188, 0.0653, 0.0173, 0.0015, 0.0051,
0.0067, 0.0055, 0.0016, 0.0033, 0.0059]
Y = gdfce, X = rgnp, P Values = [0.8134, 0.7185, 0.4338, 0.003, 0.0079, 0.0147, 0.0795, 0
.0957, 0.0919, 0.0879, 0.121, 0.0225]
Y = rgnp, X = pgnp, P Values = [0.8874, 0.8382, 0.9357, 0.029, 0.0514, 0.0632, 0.0593, 0.
1256, 0.0241, 0.0215, 0.0331, 0.0365]
Y = pgnp, X = pgnp, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.
0]
Y = ulc, X = pgnp, P Values = [0.9335, 0.9702, 0.7526, 0.8674, 0.3551, 0.4567, 0.4052, 0.
1495, 0.2284, 0.1614, 0.0502, 0.0006]
Y = gdfco, X = pgnp, P Values = [0.3587, 0.1496, 0.1966, 0.0923, 0.1296, 0.1001, 0.0663,
0.0576, 0.0838, 0.0834, 0.136, 0.2228]
Y = gdf, X = pgnp, P Values = [0.6229, 0.0018, 0.0043, 0.0001, 0.0, 0.0001, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0]
Y = gdfim, X = pgnp, P Values = [0.2488, 0.4483, 0.5697, 0.4213, 0.053, 0.0909, 0.1171, 0
.0654, 0.0898, 0.095, 0.0667, 0.0284]
Y = gdfcf, X = pgnp, P Values = [0.9123, 0.8458, 0.901, 0.1264, 0.0775, 0.1053, 0.126, 0.
0, 0.0, 0.0, 0.0, 0.0]
Y = gdfce, X = pgnp, P Values = [0.1089, 0.2287, 0.4033, 0.346, 0.4476, 0.5718, 0.3794, 0
.3011, 0.3555, 0.4314, 0.1254, 0.3264]
Y = rgnp, X = ulc, P Values = [0.351, 0.8594, 0.1999, 0.051, 0.018, 0.0029, 0.0003, 0.001
5, 0.0024, 0.0049, 0.001, 0.0095]
Y = pgnp, X = ulc, P Values = [0.5608, 0.8247, 0.8967, 0.8962, 0.8379, 0.7889, 0.632, 0.6
356, 0.7298, 0.498, 0.4533, 0.5577]
Y = ulc, X = ulc, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
Y = gdfco, X = ulc, P Values = [0.3008, 0.5683, 0.0617, 0.1396, 0.0087, 0.0064, 0.0103, 0
.004, 0.0028, 0.0006, 0.0003, 0.0006]
Y = gdf, X = ulc, P Values = [0.2972, 0.496, 0.3881, 0.4043, 0.5695, 0.6567, 0.6308, 0.75
04, 0.7411, 0.4965, 0.5956, 0.0766]
Y = gdfim, X = ulc, P Values = [0.0177, 0.0561, 0.024, 0.0208, 0.0278, 0.0205, 0.0297, 0.
0361, 0.0241, 0.0446, 0.0588, 0.0141]
Y = gdfcf, X = ulc, P Values = [0.1149, 0.4237, 0.578, 0.5765, 0.5045, 0.3306, 0.4638, 0.
5178, 0.6979, 0.7521, 0.8437, 0.8948]
Y = gdfce, X = ulc, P Values = [0.4328, 0.207, 0.0508, 0.0117, 0.0077, 0.0031, 0.0099, 0.
001, 0.0017, 0.0016, 0.0042, 0.0211]
Y = rgnp, X = gdfco, P Values = [0.4249, 0.6628, 0.1168, 0.1544, 0.1711, 0.1598, 0.1996,
0.1324, 0.1447, 0.0885, 0.0684, 0.192]
Y = pgnp, X = gdfco, P Values = [0.397, 0.6854, 0.3884, 0.2554, 0.1941, 0.2899, 0.4358, 0
.4288, 0.5205, 0.5691, 0.5262, 0.5835]
Y = ulc, X = gdfco, P Values = [0.5973, 0.6343, 0.7278, 0.774, 0.5501, 0.6437, 0.6986, 0.
6021, 0.5675, 0.6067, 0.7282, 0.5905]
Y = gdfco, X = gdfco, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0]
Y = gdf, X = gdfco, P Values = [0.0127, 0.0373, 0.0752, 0.0841, 0.1173, 0.2977, 0.397, 0.
4727, 0.2668, 0.3326, 0.4381, 0.539]
Y = gdfim, X = gdfco, P Values = [0.0395, 0.0207, 0.0488, 0.0761, 0.1605, 0.2486, 0.2987,
0.3184, 0.3326, 0.4672, 0.5688, 0.6522]
Y = gdfcf, X = gdfco, P Values = [0.8651, 0.9894, 0.9959, 0.9978, 0.9965, 0.9716, 0.9824,
0.9748, 0.834, 0.8907, 0.8476, 0.81]
Y = gdfce, X = gdfco, P Values = [0.0825, 0.1738, 0.2679, 0.2008, 0.325, 0.3548, 0.3584,
0.2826, 0.3347, 0.4483, 0.526, 0.8106]
Y = rgnp, X = gdf, P Values = [0.6251, 0.6006, 0.5653, 0.1423, 0.1611, 0.1573, 0.1312, 0.
0393, 0.0223, 0.0059, 0.0017, 0.0031]
Y = pgnp, X = gdf, P Values = [0.5071, 0.7856, 0.9158, 0.6094, 0.2431, 0.0503, 0.0409, 0.
022, 0.0233, 0.0178, 0.0225, 0.0301]
Y = ulc, X = gdf, P Values = [0.3978, 0.5388, 0.7607, 0.4529, 0.0065, 0.0027, 0.0005, 0.0
007, 0.0004, 0.0002, 0.0, 0.0]
Y = gdfco, X = gdf, P Values = [0.5362, 0.014, 0.0175, 0.0216, 0.013, 0.0146, 0.0096, 0.0
067, 0.0054, 0.0055, 0.0035, 0.0038]
Y = gdf, X = gdf, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
Y = gdfim, X = gdf, P Values = [0.0021, 0.0072, 0.0138, 0.0178, 0.0166, 0.0175, 0.0017, 0
.0058, 0.005, 0.0084, 0.0071, 0.0002]
Y = gdfcf, X = gdf, P Values = [0.5437, 0.6629, 0.8616, 0.9045, 0.8251, 0.2594, 0.1223, 0
.0664, 0.0749, 0.0887, 0.1273, 0.0498]
Y = gdfce, X = gdf, P Values = [0.0471, 0.28, 0.206, 0.0049, 0.0203, 0.0059, 0.0332, 0.02
34, 0.0128, 0.0024, 0.0119, 0.0147]
Y = rgnp, X = gdfim, P Values = [0.1783, 0.248, 0.2169, 0.2661, 0.3659, 0.1875, 0.1598, 0
.1636, 0.1956, 0.2254, 0.2324, 0.1765]
Y = pgnp, X = gdfim, P Values = [0.0001, 0.0001, 0.0001, 0.0001, 0.0002, 0.0003, 0.0008,
0.0013, 0.0022, 0.0043, 0.0061, 0.0071]
Y = ulc, X = gdfim, P Values = [0.3889, 0.6622, 0.595, 0.7422, 0.6219, 0.646, 0.4121, 0.3
083, 0.2053, 0.2659, 0.0988, 0.0861]
Y = gdfco, X = gdfim, P Values = [0.4293, 0.1697, 0.0704, 0.0008, 0.0024, 0.0024, 0.0047,
0.0009, 0.0014, 0.0019, 0.0045, 0.004]
Y = gdf, X = gdfim, P Values = [0.2089, 0.0743, 0.1885, 0.3254, 0.2965, 0.3428, 0.2698, 0
.0045, 0.0023, 0.0088, 0.0036, 0.0191]
Y = gdfim, X = gdfim, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0]
Y = gdfcf, X = gdfim, P Values = [0.663, 0.882, 0.7029, 0.5995, 0.7891, 0.2135, 0.0432, 0
.0828, 0.0069, 0.0135, 0.018, 0.0357]
Y = gdfce, X = gdfim, P Values = [0.205, 0.2204, 0.0796, 0.1715, 0.2248, 0.0113, 0.0154,
0.0118, 0.0071, 0.0117, 0.007, 0.0005]
Y = rgnp, X = gdfcf, P Values = [0.2071, 0.2219, 0.1755, 0.3, 0.2256, 0.0294, 0.0518, 0.1
181, 0.0065, 0.001, 0.0021, 0.0086]
Y = pgnp, X = gdfcf, P Values = [0.0495, 0.1199, 0.2266, 0.3605, 0.2718, 0.2698, 0.16, 0.
0738, 0.0461, 0.0719, 0.0699, 0.0997]
Y = ulc, X = gdfcf, P Values = [0.2663, 0.6717, 0.7709, 0.3423, 0.1174, 0.1056, 0.0274, 0
.0096, 0.0096, 0.0008, 0.0021, 0.0043]
Y = gdfco, X = gdfcf, P Values = [0.8147, 0.3721, 0.5836, 0.6336, 0.4311, 0.4019, 0.1012,
0.1295, 0.1901, 0.1297, 0.1616, 0.1913]
Y = gdf, X = gdfcf, P Values = [0.9323, 0.9406, 0.9648, 0.9731, 0.7979, 0.6697, 0.4835, 0
.0569, 0.0654, 0.1228, 0.1035, 0.1621]
Y = gdfim, X = gdfcf, P Values = [0.5009, 0.7402, 0.1524, 0.2627, 0.3051, 0.3644, 0.0697,
0.021, 0.0217, 0.0489, 0.0101, 0.0004]
Y = gdfcf, X = gdfcf, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0]
Y = gdfce, X = gdfcf, P Values = [0.6244, 0.8287, 0.3539, 0.4474, 0.1952, 0.3601, 0.3973,
0.0, 0.0, 0.0, 0.0, 0.0001]
Y = rgnp, X = gdfce, P Values = [0.5515, 0.5667, 0.3488, 0.0418, 0.0682, 0.1, 0.1027, 0.3
077, 0.3449, 0.3574, 0.4318, 0.1192]
Y = pgnp, X = gdfce, P Values = [0.0798, 0.1675, 0.282, 0.1903, 0.1784, 0.1436, 0.1904, 0
.2019, 0.1625, 0.2155, 0.2109, 0.2892]
Y = ulc, X = gdfce, P Values = [0.9162, 0.4988, 0.6853, 0.5728, 0.2874, 0.2159, 0.1175, 0
.1336, 0.2118, 0.0152, 0.0093, 0.0013]
Y = gdfco, X = gdfce, P Values = [0.1398, 0.0303, 0.0217, 0.0523, 0.0651, 0.1084, 0.0168,
0.0168, 0.0293, 0.0269, 0.0144, 0.001]
Y = gdf, X = gdfce, P Values = [0.0014, 0.0195, 0.0369, 0.0418, 0.04, 0.0682, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0008]
Y = gdfim, X = gdfce, P Values = [0.9894, 0.8077, 0.9259, 0.9284, 0.8799, 0.2076, 0.0556,
0.1556, 0.1714, 0.2375, 0.2381, 0.0819]
```

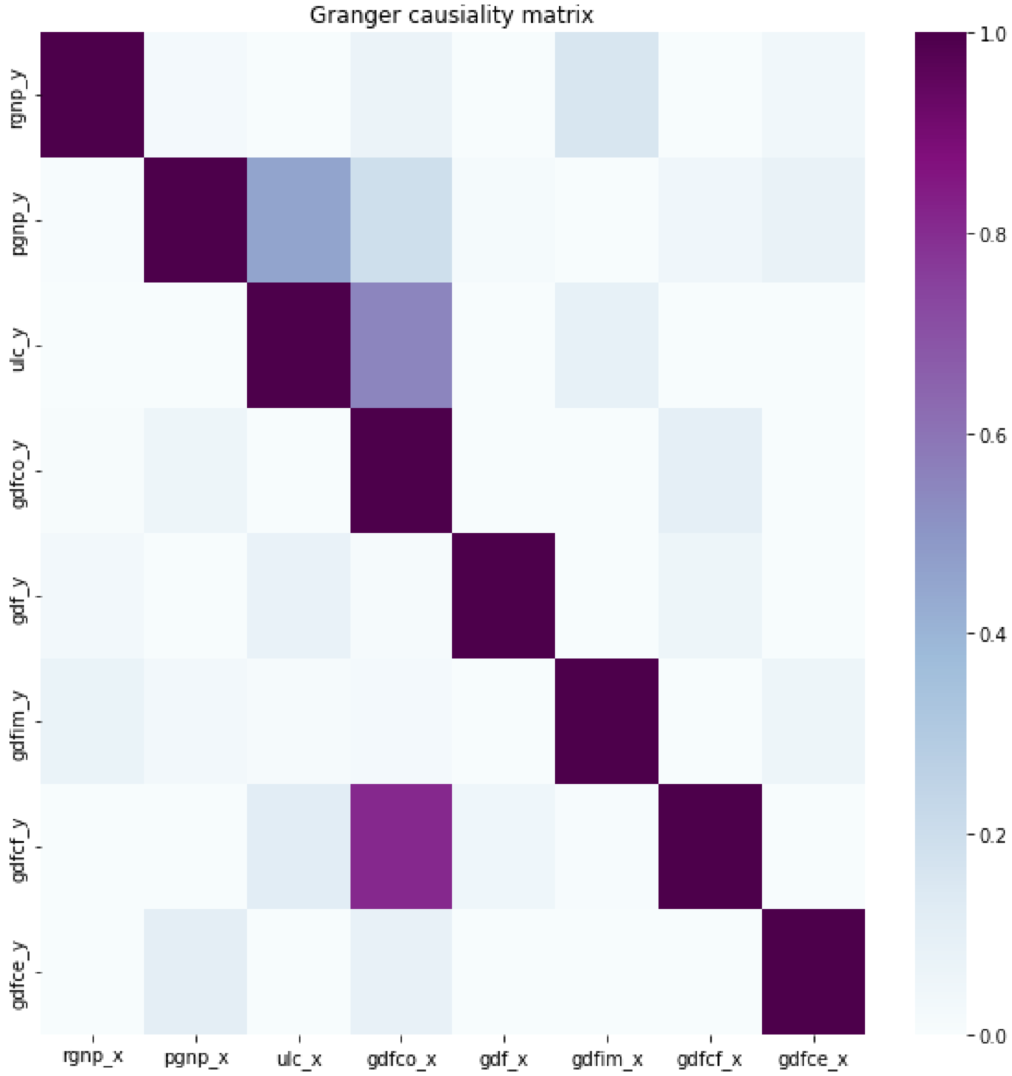
```
Y = gdfcf, X = gdfce, P Values = [0.0031, 0.0077, 0.0115, 0.0198, 0.0522, 0.0973, 0.1371,
0.1308, 0.0867, 0.0664, 0.1117, 0.16]
Y = gdfce, X = gdfce, P Values = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0]
```

Out[13]:

	rgnp_x	pgnp_x	ulc_x	gdfco_x	gdf_x	gdfim_x	gdfcf_x	gdfce_x
rgnp_y	1.0000	0.0215	0.0003	0.0684	0.0017	0.1598	0.0010	0.0418
pgnp_y	0.0070	1.0000	0.4533	0.1941	0.0178	0.0001	0.0461	0.0798
ulc_y	0.0000	0.0006	1.0000	0.5501	0.0000	0.0861	0.0008	0.0013
gdfco_y	0.0079	0.0576	0.0003	1.0000	0.0035	0.0008	0.1012	0.0010
gdf_y	0.0308	0.0000	0.0766	0.0127	1.0000	0.0023	0.0569	0.0000
gdfim_y	0.0712	0.0284	0.0141	0.0207	0.0002	1.0000	0.0004	0.0556
gdfcf_y	0.0015	0.0000	0.1149	0.8100	0.0498	0.0069	1.0000	0.0031
gdfce_y	0.0030	0.1089	0.0010	0.0825	0.0024	0.0005	0.0000	1.0000

In [17]:

```
from sklearn.metrics import ConfusionMatrixDisplay
import seaborn as sns
fig, ax = plt.subplots(1, figsize = (10,10))
disp = sns.heatmap(granger, fmt='', cmap='BuPu', ax = ax)
ax.set_title("Granger causiality matrix")
plt.show()
```



In [ ]: