appendix

October 2, 2025

0.1 Introduction

This is the notebook file to replicate our macroeconometrics approach. This notebook does not contain the `blackmarblepy` application. Source data is from `blackmarblepy` for nightlight, and BPS. If you happens to find any issues, you can find Krisna via krisna@dewanekonomi.go.id

You will see the following sections in this notebook:

1. Real GDP and quarterly night light index (NTL) graph;
2. OLS and residuals;
3. ADF test and Johansen Cointegration test;
4. VECM graph;
5. VAR graph; and
6. ARDL graph.

We do those steps for both quarterly dataset and growth dataset.

We only print the regression summary for a panel results that are best fit.

Note that you're going to need Stata to run the VAR regression. I somehow cannot get the VARMAX from statsmodels to converge unfortunately.

0.2 some Packages

[1]:
```
import pandas as pd
from pandas.tseries.offsets import QuarterEnd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from statsmodels.tsa.vector_ar.vecm import VECM, select_order, select_coint_rank
from statsmodels.tsa.api import VAR
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.ardl import ARDL
from sklearn.metrics import mean_squared_error
import statsmodels.formula.api as smf
from statsmodels.tsa.ardl import ardl_select_order
pd.options.display.max_seq_items = 4000 ## This is only for cosmetics.
```

0.3 Growth Regression

0.3.1 The GDP and night light growth dataset

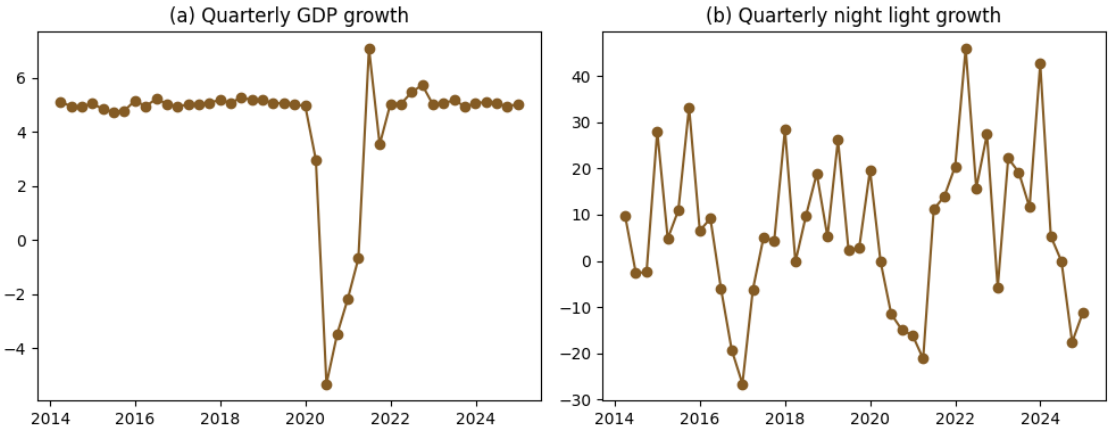
Turn on the last line of the first codeblock to see the dataframe

[2]:
```
## Data prep
### Creating data
ntl=pd.read_csv('ntl_monthly_avg_2012-2024.csv')
gdp=pd.read_excel('GDP_YoY_Quarterly_12_24.xlsx')

### Make time index
ntl.Date=pd.to_datetime(ntl['Date'])
ntl['qtr']=ntl['Date'].dt.quarter
ntl['year']=ntl['Date'].dt.year
### Averaging the radiance into quarterly, make it yoy quarterly growth
ntl=ntl.groupby(['year','qtr'])['NTL_Radiance'].mean().reset_index()
ntl['Date']=pd.date_range(start='2012-01-01', periods=len(ntl), freq='QE')
ntl=ntl[['Date','NTL_Radiance']]
ntl['g']=(gdp['Real GDP YoY Growth Indonesia'])
ntl['ntlg']=(ntl['NTL_Radiance'])
ntl['NTL_Radiancelag'] = ntl['NTL_Radiance'].shift(4)
ntl['ntlg'] = ((ntl['NTL_Radiance'] - ntl['NTL_Radiancelag']) /␣
 ↪ntl['NTL_Radiancelag']) * 100
### Creating dummy quarterly and dummy covid
ntl['q1']=np.where(ntl['Date'].dt.quarter==1,1,0)
ntl['q2']=np.where(ntl['Date'].dt.quarter==2,1,0)
ntl['q3']=np.where(ntl['Date'].dt.quarter==3,1,0)
ntl['q4']=np.where(ntl['Date'].dt.quarter==4,1,0)
ntl['covid']=np.where((ntl['Date'].dt.year>=2020) & (ntl['Date'].dt.
 ↪year<=2022),1,0)
ntl['scar']=np.where((ntl['Date'].dt.year>=2020) ,1,0)
### Back to making time index
ntl=ntl.dropna().reset_index(drop=True)
ntl['qtr']=ntl['Date'].dt.quarter
ntl['year']=ntl['Date'].dt.year
ntl['qqq']= ntl['year'].astype(str)+'q' + ntl['qtr'].astype(str) ## This is for␣
 ↪Stata time index for tsset.
ntl=ntl.set_index('Date')
ntl=ntl.asfreq('QE-DEC')
#ntlm=ntlm[['g','ntlg']]

### Creating dummy quarterly and dummy covid

## OLS-ing
mod=sm.OLS(ntl['g'], sm.add_constant(ntl['ntlg'])).fit()
ntl['resid']=mod.resid
ntl['ols']=mod.predict()
```

```
#ntl
```

0.3.2 Plot the growth dataset

[3]:
```
# Plotting GDP Growth and Night light growth side by side
fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))
ntlm=ntl[4:]
ax1.plot(ntlm['g'],color='#845B24',marker='o', linestyle='-')
ax1.set_title('(a) Quarterly GDP growth')

ax2.plot(ntlm['ntlg'], linestyle='-', color='#845B24',marker='o')
ax2.set_title('(b) Quarterly night light growth')

plt.tight_layout()
plt.savefig("fig/fig.png")
plt.show()
```



0.3.3 OLS and residuals

[4]:
```
## OLS results and plotting residuals
ntl=ntlm
print(mod.summary())
fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))

ax1.plot(ntlm['g'],color='#845B24',linestyle="-",label="observed GDP Growth")
ax1.plot(ntlm['ols'],color='#EEC051',linestyle="--",label="OLS-fitted GDP␣
↪Growth")
ax1.set_title('(a) GDP Growth')
ax1.legend()

ax2.plot(ntlm['resid'], linestyle='-', color='#845B24')
```

```
ax2.set_title('(b) OLS Residuals')

plt.tight_layout()
plt.savefig("fig/ols.png") # Turn off to not save, or change file name to save␣
  ↪in your preferred location
plt.show()
```

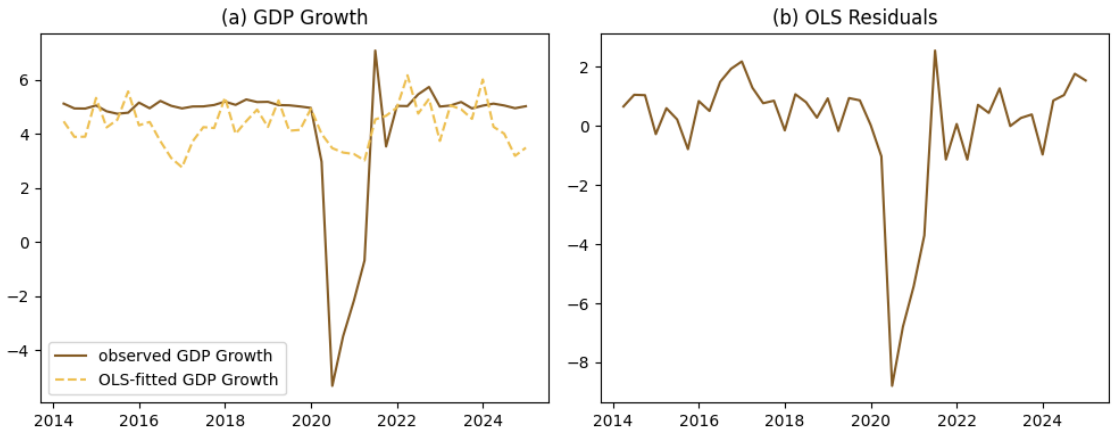
OLS Regression Results

Dep. Variable:	g	R-squared:	0.150
Model:	OLS	Adj. R-squared:	0.132
Method:	Least Squares	F-statistic:	8.126
Date:	Wed, 01 Oct 2025	Prob (F-statistic):	0.00651
Time:	22:02:59	Log-Likelihood:	-104.61
No. Observations:	48	AIC:	213.2
Df Residuals:	46	BIC:	217.0
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	4.0069	0.345	11.617	0.000	3.313	4.701
ntlg	0.0470	0.016	2.851	0.007	0.014	0.080

Omnibus:	42.129	Durbin-Watson:	0.736
Prob(Omnibus):	0.000	Jarque-Bera (JB):	126.380
Skew:	-2.449	Prob(JB):	3.61e-28
Kurtosis:	9.260	Cond. No.	22.9

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.



0.3.4 ADF test

[5]:
```python
## ADF Test for g, ntlg and OLS residuals
def adf_test(series, name=""):
    """
    Perform ADF test and print results
    """
    result = adfuller(series.dropna(), autolag="BIC")
    print(f"ADF Test for {name}")
    print(f"  Test Statistic : {result[0]:.4f}")
    print(f"  p-value        : {result[1]:.4f}")
    print(f"  #Lags Used     : {result[2]}")
    print(f"  #Observations  : {result[3]}")
    for key, value in result[4].items():
        print(f"   Critical Value {key} : {value:.4f}")
    if result[1] <= 0.05:
        print(f"  ==> {name} is stationary\n (reject H0 of unit root)\n")
    else:
        print(f"  ==> {name} is non-stationary\n (fail to reject H0)\n")

# Run ADF tests for both series
adf_test(ntlm["g"], "GDP YoY Growth")
adf_test(ntlm["ntlg"], "NTL YoY Growth")
adf_test(ntlm["resid"], "OLS Residuals")
```

```
ADF Test for GDP YoY Growth
  Test Statistic : -2.7478
  p-value        : 0.0661
  #Lags Used     : 0
  #Observations  : 43
   Critical Value 1% : -3.5925
   Critical Value 5% : -2.9315
   Critical Value 10% : -2.6041
  ==> GDP YoY Growth is non-stationary
 (fail to reject H0)

ADF Test for NTL YoY Growth
  Test Statistic : -4.2533
  p-value        : 0.0005
  #Lags Used     : 0
  #Observations  : 43
   Critical Value 1% : -3.5925
   Critical Value 5% : -2.9315
   Critical Value 10% : -2.6041
  ==> NTL YoY Growth is stationary
 (reject H0 of unit root)
```

```
ADF Test for OLS Residuals
  Test Statistic : -2.9893
  p-value        : 0.0359
  #Lags Used     : 0
  #Observations  : 43
   Critical Value 1% : -3.5925
   Critical Value 5% : -2.9315
   Critical Value 10% : -2.6041
  ==> OLS Residuals is stationary
 (reject H0 of unit root)
```

0.3.5 Johansen Cointegration test

[6]:
```
# Select optimal lag order
ntlm=ntl[['g','ntlg']]
lag_order = select_order(ntlm.asfreq('QE-DEC'), maxlags=12, deterministic="ci")
print(lag_order.summary())

# Select cointegration rank
coint_rank = select_coint_rank(ntlm.asfreq('QE-DEC'), det_order=0,␣
 ↪k_ar_diff=lag_order.bic)
print(coint_rank.summary())
```

```
 VECM Order Selection (* highlights the minimums)
=================================================
       AIC         BIC         FPE         HQIC
-------------------------------------------------
0        7.167      7.537*      1300.*       7.288
1        7.384       7.939       1627.       7.565
2        7.552       8.292       1950.       7.793
3        7.303       8.228       1557.       7.604
4        7.103       8.214       1324.       7.465
5        7.182       8.477       1513.       7.604
6        7.367       8.847       1972.       7.849
7        7.216       8.882       1896.       7.759
8        7.469       9.320       2854.       8.072
9        7.035       9.070       2303.       7.698
10       6.595       8.815       2040.       7.318
11       6.339       8.745       2570.       7.123
12      5.159*       7.750       1815.      6.004*
-------------------------------------------------
Johansen cointegration test using trace test statistic with 5% significance
level
=====================================
r_0 r_1 test statistic critical value
-------------------------------------
```

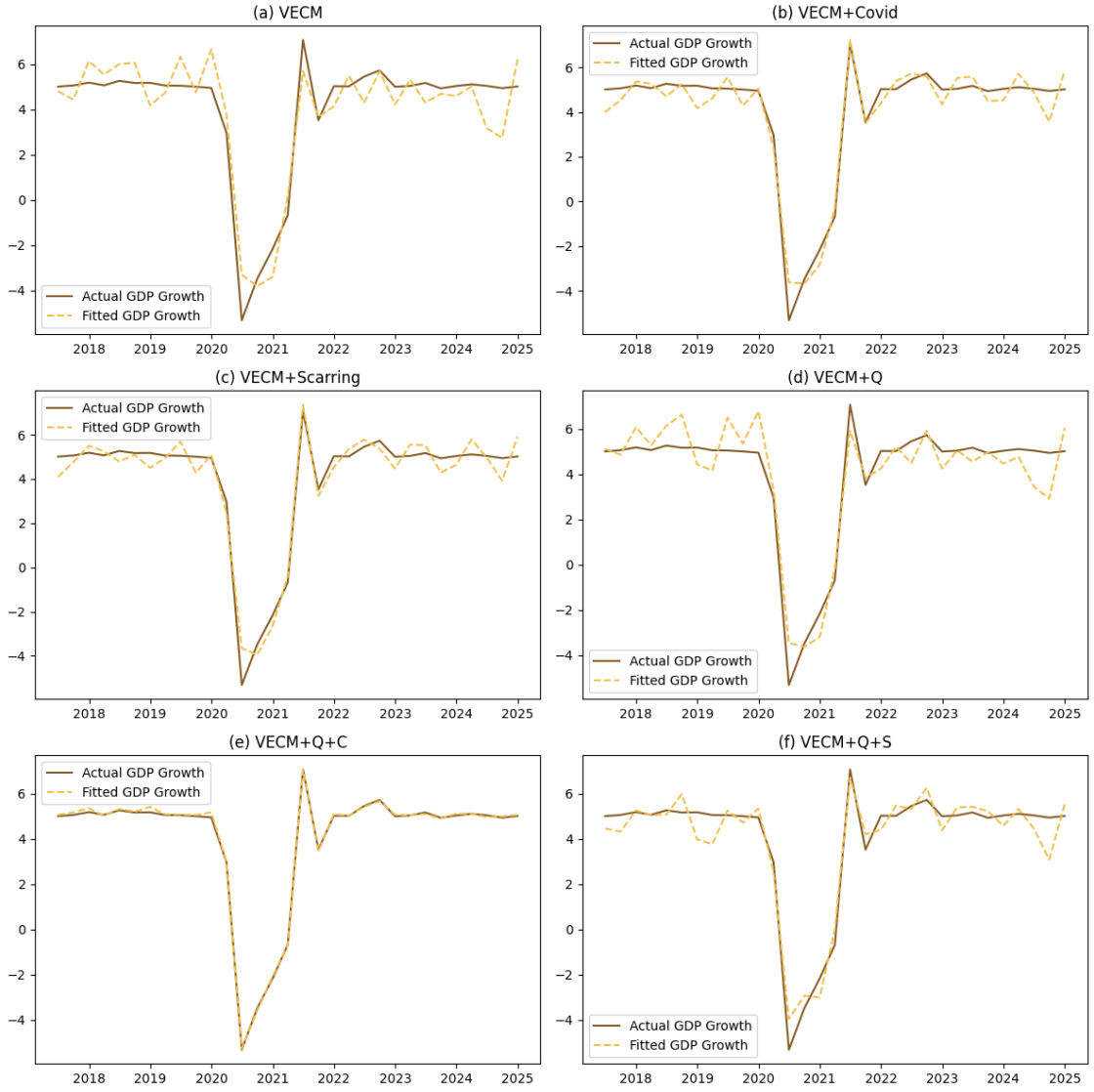
```
  0   2          27.82          15.49
  1   2          7.259          3.841
-------------------------------------
```

0.3.6 VECM with growth dataset

Note that I save the graph in a folder called `fig`. Please create the folder first before running the code. If you're not planning to save the graph, you can comment out the `plt.savefig` line.

[7]:
```
en=ntl[['g','ntlg']]
exc=ntl[['covid']]
exs=ntl[['scar']]
exq=ntl[['q1','q2','q3']]
exqc=ntl[['q1','q2','q3','covid']]
exqs=ntl[['q1','q2','q3','scar']]

lag=lag_order.aic

ve = VECM(en,k_ar_diff=lag, coint_rank=1, deterministic="ci").fit()
vec = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exc,deterministic="ci").fit()
ves = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exs,deterministic="ci").fit()
veq = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exq,deterministic="ci").fit()
veqc = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exqc,deterministic="ci").fit()
veqs = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exqs,deterministic="ci").fit()

models = {'fve': ve, 'fvec': vec,'fves': ves, 'fveq': veq,'fveqc': veqc,␣
 ↪'fveqs': veqs}
results = {}

for name, model in models.items():
    fitted = pd.DataFrame(model.fittedvalues, columns=en.columns)
    fitted.index = pd.date_range(end='2024-12-31', periods=31, freq='QE')
    merged = pd.merge(en, fitted, left_index=True, right_index=True,␣
 ↪suffixes=('', f'_fitted'))
    results[name] = merged


fig, ax = plt.subplots(3,2,figsize=(12, 12))

ax[0,0].plot(results['fve']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,0].plot(results['fve']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[0,0].set_title('(a) VECM')
ax[0,0].legend()
```

```
ax[0,1].plot(results['fvec']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,1].plot(results['fvec']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[0,1].set_title('(b) VECM+Covid')
ax[0,1].legend()

ax[1,0].plot(results['fves']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,0].plot(results['fves']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,0].set_title('(c) VECM+Scarring')
ax[1,0].legend()

ax[1,1].plot(results['fveq']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,1].plot(results['fveq']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,1].set_title('(d) VECM+Q')
ax[1,1].legend()

ax[2,0].plot(results['fveqc']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,0].plot(results['fveqc']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,0].set_title('(e) VECM+Q+C')
ax[2,0].legend()

ax[2,1].plot(results['fveqs']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,1].plot(results['fveqs']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,1].set_title('(f) VECM+Q+S')
ax[2,1].legend()
plt.tight_layout()
plt.savefig("fig/VECM.png")
plt.show()
```



[8]:
```
print(veqc.summary())
```

```
Det. terms outside the coint. relation & lagged endog. parameters for equation g
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
exog1        -13.7498      0.227    -60.601      0.000     -14.195     -13.305
exog2         -2.7412      0.064    -42.647      0.000      -2.867      -2.615
exog3          5.8433      0.099     58.801      0.000       5.649       6.038
exog4         68.6318      1.148     59.794      0.000      66.382      70.881
L1.g         -15.5121      0.264    -58.768      0.000     -16.029     -14.995
L1.ntlg       -3.4491      0.059    -58.655      0.000      -3.564      -3.334
L2.g         -19.1633      0.325    -59.010      0.000     -19.800     -18.527
```

L2.ntlg	-4.5342	0.075	-60.759	0.000	-4.680	-4.388
L3.g	0.6169	0.032	18.982	0.000	0.553	0.681
L3.ntlg	-4.2461	0.072	-59.303	0.000	-4.386	-4.106
L4.g	-8.4759	0.144	-59.006	0.000	-8.757	-8.194
L4.ntlg	-5.3860	0.089	-60.210	0.000	-5.561	-5.211
L5.g	-10.4198	0.180	-57.801	0.000	-10.773	-10.066
L5.ntlg	-1.7644	0.031	-57.153	0.000	-1.825	-1.704
L6.g	-9.9096	0.169	-58.518	0.000	-10.241	-9.578
L6.ntlg	-3.9247	0.064	-60.903	0.000	-4.051	-3.798
L7.g	13.6827	0.216	63.422	0.000	13.260	14.106
L7.ntlg	-4.5100	0.074	-60.714	0.000	-4.656	-4.364
L8.g	-4.7570	0.081	-58.519	0.000	-4.916	-4.598
L8.ntlg	-4.1480	0.068	-61.272	0.000	-4.281	-4.015
L9.g	0.4382	0.025	17.493	0.000	0.389	0.487
L9.ntlg	-0.2571	0.006	-40.641	0.000	-0.269	-0.245
L10.g	-3.1649	0.060	-52.575	0.000	-3.283	-3.047
L10.ntlg	-2.4267	0.040	-61.389	0.000	-2.504	-2.349
L11.g	9.5875	0.154	62.396	0.000	9.286	9.889
L11.ntlg	-2.6394	0.044	-60.432	0.000	-2.725	-2.554
L12.g	-0.2446	0.018	-13.615	0.000	-0.280	-0.209
L12.ntlg	-3.8172	0.061	-62.070	0.000	-3.938	-3.697

Det. terms outside the coint. relation & lagged endog. parameters for equation ntlg

	coef	std err	z	P>|z|	[0.025	0.975]
exog1	33.5604	7.749	4.331	0.000	18.372	48.748
exog2	1.7441	2.195	0.794	0.427	-2.559	6.047
exog3	-11.8698	3.394	-3.497	0.000	-18.522	-5.218
exog4	-158.4618	39.201	-4.042	0.000	-235.294	-81.629
L1.g	40.9036	9.015	4.537	0.000	23.235	58.573
L1.ntlg	8.2881	2.008	4.127	0.000	4.352	12.224
L2.g	44.9582	11.091	4.054	0.000	23.220	66.696
L2.ntlg	10.0932	2.549	3.960	0.000	5.098	15.089
L3.g	9.6145	1.110	8.662	0.000	7.439	11.790
L3.ntlg	9.9316	2.445	4.061	0.000	5.139	14.724
L4.g	25.4385	4.906	5.185	0.000	15.823	35.054
L4.ntlg	10.9218	3.055	3.575	0.000	4.934	16.910
L5.g	27.7443	6.157	4.506	0.000	15.677	39.811
L5.ntlg	4.7112	1.054	4.468	0.000	2.645	6.778
L6.g	21.2026	5.784	3.666	0.000	9.867	32.538
L6.ntlg	8.6960	2.201	3.951	0.000	4.382	13.010
L7.g	-19.2247	7.368	-2.609	0.009	-33.666	-4.783
L7.ntlg	9.8645	2.537	3.888	0.000	4.892	14.837
L8.g	11.8049	2.776	4.252	0.000	6.363	17.246
L8.ntlg	7.9601	2.312	3.443	0.001	3.428	12.492
L9.g	1.4561	0.856	1.702	0.089	-0.221	3.133
L9.ntlg	1.3154	0.216	6.089	0.000	0.892	1.739

```
L10.g          6.9053      2.056      3.359      0.001       2.876      10.935
L10.ntlg       5.1242      1.350      3.795      0.000       2.478       7.770
L11.g        -14.4932      5.248     -2.762      0.006     -24.779      -4.208
L11.ntlg       5.7904      1.492      3.882      0.000       2.867       8.714
L12.g          1.5383      0.614      2.507      0.012       0.336       2.741
L12.ntlg       7.3177      2.100      3.484      0.000       3.201      11.434
                Loading coefficients (alpha) for equation g
================================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
ec1           21.0690      0.357     58.965      0.000      20.369      21.769
               Loading coefficients (alpha) for equation ntlg
================================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
ec1          -52.7266     12.203     -4.321      0.000     -76.645     -28.808
          Cointegration relations for loading-coefficients-column 1
================================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
beta.1         1.0000          0          0      0.000       1.000       1.000
beta.2         0.1978        nan        nan        nan         nan         nan
const         -6.0585        nan        nan        nan         nan         nan
================================================================================

c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\vector_ar\vecm.py:1582: RuntimeWarning: invalid value
encountered in sqrt
  last_rows_1d = np.sqrt(np.diag(mat1.dot(mat2)))
```

0.3.7 VAR with growth dataset

Please note that I am using Stata to run the VAR regression. I call Stata to this notebook using magic command, and `stata_setup` and `pystata` package have to be installed prior. See this documentation for more information. Obviously you can just run this code in Stata directly. Running Stata on a Jupyter Notebook can be a bit clunky. If you are not very familiar with running Stata on Jupyter Notebook, I suggest you to run the VAR regression directly in Stata.

VAR results are not as good as VECM results. I include this section just for the sake of completeness.

[9]:
```
import stata_setup
stata_setup.config("C:/program files/statanow19/","mp") # Change this to your␣
 ↪Stata installation directory
## If this line runs correctly, you will see the Stata logo printed out in the␣
 ↪output just like on the Stata IDE.
```

```
  ___  ____  ____  ____  ____ ®
```

```
 /__    /   ____/   /   ____/      StataNow 19.5
___/   /   /___/   /   /___/       MP-Parallel Edition

 Statistics and Data Science       Copyright 1985-2025 StataCorp LLC
                                   StataCorp
                                   4905 Lakeway Drive
                                   College Station, Texas 77845 USA
                                   800-782-8272        https://www.stata.com
                                   979-696-4600        service@stata.com

Stata license: Single-user 4-core , expiring 24 Dec 2025
Serial number: 501909221392
  Licensed to: Krisna Gupta
               Politeknik APP Jakarta

Notes:
      1. Unicode is supported; see help unicode_advice.
      2. More than 2 billion observations are allowed; see help obs_advice.
      3. Maximum number of variables is set to 5,000 but can be increased;
          see help set_maxvar.
```

[10]:
```
%%stata -d ntl
gen qdate = quarterly(qqq, "YQ")
format qdate %tq
tsset qdate
```

```
. gen qdate = quarterly(qqq, "YQ")

. format qdate %tq

. tsset qdate

Time variable: qdate, 2014q1 to 2024q4
        Delta: 1 quarter

.
```

[11]:
```
%%stata
var g ntlg, lags(1/8)
predict r_g
tsline g r_g, xtitle("") title("(a) VAR") lcolor("#845B24" "#EEC051")␣
 ↪lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2␣
 ↪"Predicted")) lwidth(0.6 0.6)
graph save g1.gph, replace
var g ntlg, lags(1/8) exog(covid)
drop r_g
predict r_g
```

```
tsline g r_g, xtitle("") title("(b) VAR+Covid") lcolor("#845B24" "#EEC051")␣
  ↪lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2␣
  ↪"Predicted")) lwidth(0.6 0.6)
graph save g2.gph, replace
var g ntlg, lags(1/8) exog(scar)
drop r_g
predict r_g
tsline g r_g, xtitle("") title("(c) VAR+Scarring") lcolor("#845B24" "#EEC051")␣
  ↪lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2␣
  ↪"Predicted")) lwidth(0.6 0.6)
graph save g3.gph, replace
var g ntlg, lags(1/8) exog(q1 q2 q3)
drop r_g
predict r_g
tsline g r_g, xtitle("") title("(d) VAR+Quarter") lcolor("#845B24" "#EEC051")␣
  ↪lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2␣
  ↪"Predicted")) lwidth(0.6 0.6)
graph save g4.gph, replace
var g ntlg, lags(1/8) exog(q1 q2 q3 covid)
drop r_g
predict r_g
tsline g r_g, xtitle("") title("(e) VAR+Q+C") lcolor("#845B24" "#EEC051")␣
  ↪lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2␣
  ↪"Predicted")) lwidth(0.6 0.6)
graph save g5.gph, replace
var g ntlg, lags(1/8) exog(q1 q2 q3 scar)
drop r_g
predict r_g
tsline g r_g, xtitle("") title("(f) VAR+Q+S") lcolor("#845B24" "#EEC051")␣
  ↪lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2␣
  ↪"Predicted")) lwidth(0.6 0.6)
graph save g6.gph, replace

graph combine g1.gph g2.gph g3.gph g4.gph g5.gph g6.gph, rows(3) cols(2)␣
  ↪imargin(2 2 2 2)
graph export "fig/VARstata.png", width(2400) height(3200) replace
```

```
. var g ntlg, lags(1/8)

Vector autoregression

Sample: 2016q1 thru 2024q4                      Number of obs     =         36
Log likelihood =  -194.5246                     AIC               =   12.69581
FPE            =   1316.685                     HQIC              =   13.21779
Det(Sigma_ml)  =   169.2144                     SBIC              =   14.19136
```

Equation	Parms	RMSE	R-sq	chi2	P>chi2
g	17	2.14914	0.6468	65.92458	0.0000
ntlg	17	12.7098	0.7111	88.63131	0.0000

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
g						
g						
L1.	.8663179	.1783422	4.86	0.000	.5167735	1.215862
L2.	.0997438	.2240817	0.45	0.656	-.3394483	.5389358
L3.	-.2453768	.2366989	-1.04	0.300	-.7092982	.2185446
L4.	-.4127796	.2229512	-1.85	0.064	-.849756	.0241967
L5.	.6120339	.213067	2.87	0.004	.1944304	1.029637
L6.	-.2124892	.2352451	-0.90	0.366	-.673561	.2485827
L7.	-.1166599	.2447776	-0.48	0.634	-.5964151	.3630954
L8.	.0061414	.1973451	0.03	0.975	-.3806479	.3929307
ntlg						
L1.	-.0104131	.0256126	-0.41	0.684	-.0606129	.0397866
L2.	-.0177145	.0271547	-0.65	0.514	-.0709367	.0355077
L3.	.0241892	.0261618	0.92	0.355	-.027087	.0754654
L4.	.0140027	.0260909	0.54	0.591	-.0371345	.06514
L5.	-.0440321	.0261414	-1.68	0.092	-.0952683	.0072041
L6.	.0135243	.0273019	0.50	0.620	-.0399865	.067035
L7.	-.0015925	.0245472	-0.06	0.948	-.0497042	.0465191
L8.	.0020723	.0239029	0.09	0.931	-.0447766	.0489211
_cons	1.799064	.8364123	2.15	0.031	.1597264	3.438402
ntlg						
g						
L1.	1.594895	1.054701	1.51	0.130	-.4722801	3.662071
L2.	-2.074009	1.3252	-1.57	0.118	-4.671354	.5233348
L3.	2.67042	1.399817	1.91	0.056	-.0731709	5.414012
L4.	-2.642794	1.318514	-2.00	0.045	-5.227035	-.0585537
L5.	2.559623	1.26006	2.03	0.042	.0899513	5.029295
L6.	-4.200308	1.391219	-3.02	0.003	-6.927048	-1.473569
L7.	2.853991	1.447594	1.97	0.049	.0167592	5.691223
L8.	-1.453177	1.167082	-1.25	0.213	-3.740616	.8342614
ntlg						
L1.	.4852617	.1514705	3.20	0.001	.1883849	.7821385
L2.	.107001	.1605905	0.67	0.505	-.2077506	.4217527
L3.	.0788323	.1547187	0.51	0.610	-.2244109	.3820755
L4.	-.7341223	.1542995	-4.76	0.000	-1.036544	-.4317009

```
         L5. |   .2344589   .1545979     1.52   0.129    -.0685475    .5374654
         L6. |   .1423958   .1614611     0.88   0.378    -.1740621    .4588537
         L7. |   .2404539     .14517     1.66   0.098    -.0440741    .5249819
         L8. |   -.429755   .1413598    -3.04   0.002    -.7068152   -.1526948
             |
       _cons |    8.35872    4.94647     1.69   0.091    -1.336184    18.05362
------------------------------------------------------------------------------

. predict r_g
(option xb assumed; fitted values)
(8 missing values generated)

. tsline g r_g, xtitle("") title("(a) VAR") lcolor("#845B24" "#EEC051") lpatter
> n(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2 "Predict
> ed")) lwidth(0.6 0.6)

. graph save g1.gph, replace
file g1.gph saved

. var g ntlg, lags(1/8) exog(covid)

Vector autoregression

Sample: 2016q1 thru 2024q4                      Number of obs     =         36
Log likelihood =  -176.8813                     AIC               =   11.82674
FPE            =   571.4691                     HQIC              =   12.37943
Det(Sigma_ml)  =   63.49656                     SBIC              =   13.41026

Equation           Parms      RMSE     R-sq      chi2     P>chi2
------------------------------------------------------------------
g                    18     1.35669   0.8667   233.9757   0.0000
ntlg                 18      12.061   0.7536   110.0895   0.0000
------------------------------------------------------------------

------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
g            |
           g |
         L1. |   .1748662   .1416427     1.23   0.217    -.1027485    .4524808
         L2. |   .1876639   .1381561     1.36   0.174    -.0831171     .458445
         L3. |  -.2937638   .1455721    -2.02   0.044    -.5790798   -.0084478
         L4. |  -.7381607    .143352    -5.15   0.000    -1.019125    -.457196
         L5. |   .2962851   .1371812     2.16   0.031     .0274148    .5651554
         L6. |   .0453428   .1483668     0.31   0.760    -.2454508    .3361364
         L7. |  -.5001959   .1584252    -3.16   0.002    -.8107035   -.1896883
         L8. |   -.265034   .1262614    -2.10   0.036    -.5125017   -.0175662
             |
```

```
        ntlg |
          L1. |   .0222343   .0162978     1.36   0.172    -.0097088    .0541775
          L2. |  -.0219189   .0166938    -1.31   0.189     -.054638    .0108003
          L3. |   .0011994   .0163494     0.07   0.942    -.0308448    .0332436
          L4. |   .0280034   .0161339     1.74   0.083    -.0036184    .0596253
          L5. |  -.0170172   .0164405    -1.04   0.301      -.04924    .0152057
          L6. |  -.0052964   .0169522    -0.31   0.755    -.0385222    .0279293
          L7. |  -.0000298   .0150841    -0.00   0.998    -.0295941    .0295344
          L8. |  -.0012635   .0146932    -0.09   0.931    -.0300617    .0275347
              |
       covid |  -5.749207   .7462348    -7.70   0.000      -7.2118   -4.286613
       _cons |   10.56008   1.247899     8.46   0.000     8.114245    13.00592
-------------+----------------------------------------------------------------
ntlg         |
           g |
          L1. |  -.3914979   1.259204    -0.31   0.756    -2.859493    2.076497
          L2. |  -1.821434   1.228209    -1.48   0.138    -4.228679    .5858115
          L3. |   2.531415   1.294136     1.96   0.050    -.0050458    5.067876
          L4. |  -3.577545     1.2744    -2.81   0.005    -6.075323   -1.079767
          L5. |   1.652544   1.219542     1.36   0.175    -.7377142    4.042802
          L6. |  -3.459612   1.318982    -2.62   0.009    -6.044769   -.8744555
          L7. |   1.752174     1.4084     1.24   0.213     -1.00824    4.512588
          L8. |  -2.232207   1.122464    -1.99   0.047    -4.432196   -.0322169
              |
        ntlg |
          L1. |    .579051   .1448879     4.00   0.000     .2950759    .8630261
          L2. |   .0949228   .1484076     0.64   0.522    -.1959508    .3857964
          L3. |   .0127875   .1453463     0.09   0.930     -.272086     .297661
          L4. |  -.6939012   .1434304    -4.84   0.000    -.9750196   -.4127829
          L5. |   .3120671   .1461563     2.14   0.033      .025606    .5985282
          L6. |    .088328   .1507054     0.59   0.558    -.2070491    .3837051
          L7. |   .2449432   .1340974     1.83   0.068    -.0178828    .5077693
          L8. |  -.4393379   .1306227    -3.36   0.001    -.6953538   -.1833221
              |
       covid |  -16.51624   6.634031    -2.49   0.013    -29.51871   -3.513781
       _cons |   33.52725   11.09383     3.02   0.003     11.78375    55.27076
------------------------------------------------------------------------------

. drop r_g

. predict r_g
(option xb assumed; fitted values)
(8 missing values generated)

. tsline g r_g, xtitle("") title("(b) VAR+Covid") lcolor("#845B24" "#EEC051") l
> pattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2 "P
> redicted")) lwidth(0.6 0.6)
```

```
. graph save g2.gph, replace
file g2.gph saved

. var g ntlg, lags(1/8) exog(scar)

Vector autoregression

Sample: 2016q1 thru 2024q4                      Number of obs     =         36
Log likelihood =  -189.8773                     AIC               =   12.54874
FPE            =   1176.393                     HQIC              =   13.10143
Det(Sigma_ml)  =   130.7103                     SBIC              =   14.13226

Equation           Parms      RMSE     R-sq      chi2     P>chi2
------------------------------------------------------------------
g                     18     2.06551   0.6909   80.47556   0.0000
ntlg                  18     12.9355   0.7165   91.00461   0.0000
------------------------------------------------------------------

------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
g            |
           g |
         L1. |   .7040603   .1815356     3.88   0.000     .3482571    1.059864
         L2. |   .0867605   .2096962     0.41   0.679    -.3242365    .4977574
         L3. |  -.2524857    .221443    -1.14   0.254     -.686506    .1815346
         L4. |  -.5769612   .2207766    -2.61   0.009    -1.009675   -.1442471
         L5. |   .5367678   .2020604     2.66   0.008     .1407367    .9327989
         L6. |  -.2136082   .2200613    -0.97   0.332    -.6449204     .217704
         L7. |  -.1776302    .230552    -0.77   0.441    -.6295038    .2742434
         L8. |  -.1008588   .1905452    -0.53   0.597    -.4743206    .2726029
             |
        ntlg |
         L1. |   .0059384   .0250215     0.24   0.812    -.0431028    .0549796
         L2. |  -.0119651   .0255282    -0.47   0.639    -.0619995    .0380693
         L3. |   .0308067   .0246466     1.25   0.211    -.0174998    .0791132
         L4. |   .0339879   .0259501     1.31   0.190    -.0168733    .0848492
         L5. |  -.0319976   .0250236    -1.28   0.201    -.0810429    .0170478
         L6. |   .0165408   .0255743     0.65   0.518    -.0335839    .0666655
         L7. |   .0144886   .0240334     0.60   0.547    -.0326161    .0615932
         L8. |   .0155112   .0231325     0.67   0.503    -.0298277    .0608502
             |
        scar |  -1.959583    .864382    -2.27   0.023     -3.65374   -.2654249
       _cons |   4.668403   1.487997     3.14   0.002     1.751983    7.584822
-------------+----------------------------------------------------------------
ntlg         |
           g |
         L1. |   1.966019    1.13689     1.73   0.084    -.2622455    4.194283
```

```
         L2. |  -2.044313    1.31325    -1.56   0.120    -4.618235    .5296086
         L3. |    2.68668   1.386816     1.94   0.053    -.0314283    5.404789
         L4. |   -2.26727   1.382642    -1.64   0.101    -4.977199    .4426585
         L5. |   2.731775    1.26543     2.16   0.031      .251579    5.211972
         L6. |  -4.197749   1.378163    -3.05   0.002    -6.898898     -1.4966
         L7. |   2.993445   1.443862     2.07   0.038     .1635275    5.823363
         L8. |  -1.208441   1.193314    -1.01   0.311    -3.547294    1.130411
             |
        ntlg |
         L1. |   .4478616   .1567003     2.86   0.004     .1407346    .7549886
         L2. |   .0938507   .1598739     0.59   0.557    -.2194963    .4071978
         L3. |   .0636964   .1543527     0.41   0.680    -.2388293    .3662221
         L4. |  -.7798334   .1625159    -4.80   0.000    -1.098359    -.461308
         L5. |    .206933   .1567136     1.32   0.187    -.1002199     .514086
         L6. |   .1354962   .1601623     0.85   0.398    -.1784161    .4494086
         L7. |   .2036724   .1505124     1.35   0.176    -.0913266    .4986713
         L8. |  -.4604932   .1448704    -3.18   0.001    -.7444341   -.1765524
             |
        scar |   4.482049   5.413305     0.83   0.408    -6.127834    15.09193
       _cons |   1.795834   9.318773     0.19   0.847    -16.46863    20.06029
------------------------------------------------------------------------------

. drop r_g

. predict r_g
(option xb assumed; fitted values)
(8 missing values generated)

. tsline g r_g, xtitle("") title("(c) VAR+Scarring") lcolor("#845B24" "#EEC051"
> ) lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2
>  "Predicted")) lwidth(0.6 0.6)

. graph save g3.gph, replace
file g3.gph saved

. var g ntlg, lags(1/8) exog(q1 q2 q3)

Vector autoregression

Sample: 2016q1 thru 2024q4                      Number of obs     =         36
Log likelihood =  -191.6514                     AIC               =   12.86952
FPE            =   1767.061                     HQIC              =   13.48362
Det(Sigma_ml)  =   144.2499                     SBIC              =   14.62899

Equation           Parms      RMSE     R-sq      chi2     P>chi2
------------------------------------------------------------------
g                     20     2.30201   0.6587   69.49338   0.0000
ntlg                  20     13.0642   0.7430   104.0788   0.0000
```

```
------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
g            |
           g |
         L1. |   .8962666    .177863     5.04   0.000     .5476616    1.244872
         L2. |   .0401729   .2293531     0.18   0.861     -.409351    .4896968
         L3. |  -.1990741   .2388743    -0.83   0.405    -.6672591    .2691109
         L4. |  -.4328902   .2201139    -1.97   0.049    -.8643056   -.0014748
         L5. |   .6254389   .2102942     2.97   0.003       .21327    1.037608
         L6. |  -.2399995   .2334913    -1.03   0.304    -.6976341    .2176351
         L7. |   -.076091   .2442846    -0.31   0.755    -.5548799     .402698
         L8. |  -.0146556   .1955946    -0.07   0.940    -.3980139    .3687028
             |
        ntlg |
         L1. |  -.0103255   .0256241    -0.40   0.687    -.0605477    .0398967
         L2. |  -.0138605   .0269382    -0.51   0.607    -.0666584    .0389373
         L3. |   .0209709   .0261749     0.80   0.423    -.0303308    .0722727
         L4. |    .014378   .0258757     0.56   0.578    -.0363375    .0650935
         L5. |  -.0450308   .0258525    -1.74   0.082    -.0957007     .005639
         L6. |   .0181309    .027204     0.67   0.505    -.0351879    .0714497
         L7. |  -.0076532   .0252032    -0.30   0.761    -.0570507    .0417442
         L8. |   .0058271   .0241967     0.24   0.810    -.0415977    .0532518
             |
          q1 |  -.5793154   .7797105    -0.74   0.457     -2.10752    .9488892
          q2 |  -.2083088    .748542    -0.28   0.781    -1.675424    1.258807
          q3 |  -.7893143    .754274    -1.05   0.295    -2.267664    .6890355
       _cons |   2.167526   .9281164     2.34   0.020     .3484515    3.986601
-------------+----------------------------------------------------------------
ntlg         |
           g |
         L1. |   1.574456   1.009395     1.56   0.119    -.4039209    3.552834
         L2. |  -1.996616   1.301608    -1.53   0.125    -4.547721    .5544888
         L3. |   2.480235   1.355642     1.83   0.067     -.176774    5.137244
         L4. |  -2.627243   1.249174    -2.10   0.035     -5.07558   -.1789066
         L5. |   2.633115   1.193446     2.21   0.027     .2940039    4.972226
         L6. |  -4.127914   1.325093    -3.12   0.002    -6.725048    -1.53078
         L7. |   2.785948   1.386346     2.01   0.044     .0687603    5.503136
         L8. |  -1.421732   1.110024    -1.28   0.200    -3.597339    .7538747
             |
        ntlg |
         L1. |   .4941087   .1454198     3.40   0.001     .2090912    .7791262
         L2. |   .1330393   .1528776     0.87   0.384    -.1665953    .4326739
         L3. |   .0866928   .1485456     0.58   0.559    -.2044512    .3778368
         L4. |  -.7689953   .1468481    -5.24   0.000    -1.056812   -.4811782
         L5. |   .2261249   .1467159     1.54   0.123    -.0614331    .5136828
```

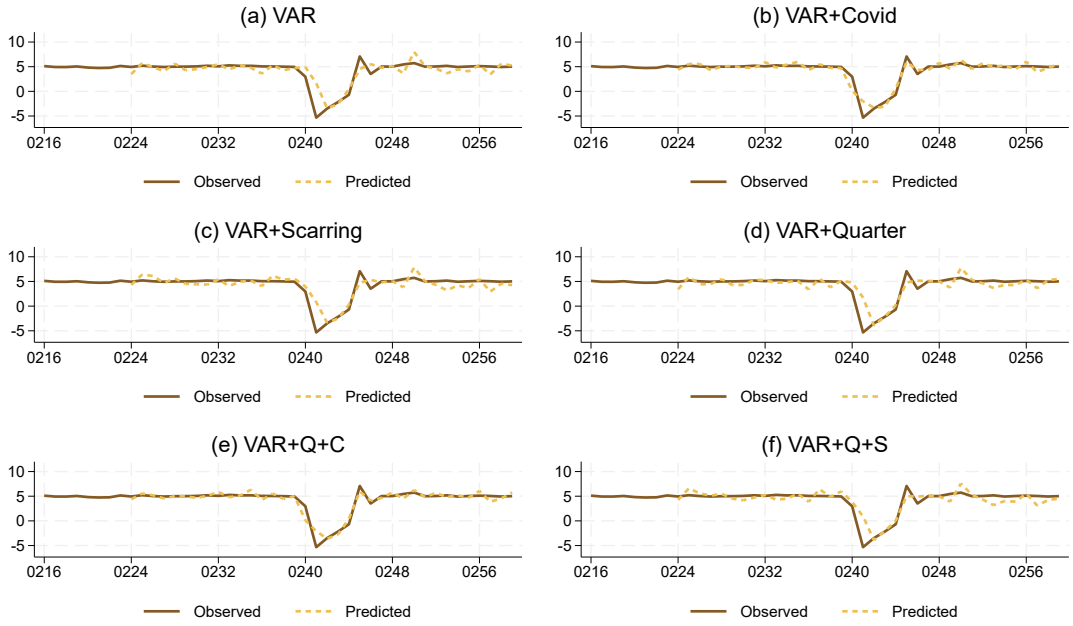
```
         L6. |   .1711385   .1543859     1.11   0.268    -.1314524    .4737294
         L7. |   .2595371   .1430315     1.81   0.070    -.0207994    .5398736
         L8. |  -.4628727   .1373195    -3.37   0.001    -.7320139   -.1937315
             |
          q1 |  -.6163833   4.424956    -0.14   0.889    -9.289137     8.05637
          q2 |  -6.727303    4.24807    -1.58   0.113    -15.05337    1.598761
          q3 |  -6.343294     4.2806    -1.48   0.138    -14.73312    2.046528
       _cons |   11.71167   5.267178     2.22   0.026      1.38819    22.03515
------------------------------------------------------------------------------

. drop r_g

. predict r_g
(option xb assumed; fitted values)
(8 missing values generated)

. tsline g r_g, xtitle("") title("(d) VAR+Quarter") lcolor("#845B24" "#EEC051")
>  lpattern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2
> "Predicted")) lwidth(0.6 0.6)

. graph save g4.gph, replace
file g4.gph saved

. var g ntlg, lags(1/8) exog(q1 q2 q3 covid)

Vector autoregression

Sample: 2016q1 thru 2024q4                      Number of obs     =         36
Log likelihood =  -173.6235                     AIC               =   11.97908
FPE            =   765.0942                     HQIC              =   12.62389
Det(Sigma_ml)  =   52.98437                     SBIC              =   13.82652

Equation           Parms      RMSE     R-sq      chi2     P>chi2
-------------------------------------------------------------
g                     21     1.45277   0.8726   246.5356   0.0000
ntlg                  21     12.2611   0.7878    133.632   0.0000
-------------------------------------------------------------

------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
g            |
           g |
         L1. |   .1960951   .1411607     1.39   0.165    -.0805748    .4727651
         L2. |   .1551976   .1409251     1.10   0.271    -.1210106    .4314057
         L3. |  -.2740683   .1462824    -1.87   0.061    -.5607766    .0126399
         L4. |  -.7451766   .1403728    -5.31   0.000    -1.020302   -.4700509
         L5. |   .3097424    .134766     2.30   0.022     .0456059     .573879
```

```
          L6. |   .0291779   .1468174     0.20   0.842    -.2585788    .3169346
          L7. |  -.4772831   .1579417    -3.02   0.003    -.7868432   -.1677231
          L8. |  -.2713279   .1239958    -2.19   0.029    -.5143552   -.0283005
              |
        ntlg |
          L1. |   .0228612   .0162292     1.41   0.159    -.0089476    .0546699
          L2. |  -.0190394    .016474    -1.16   0.248    -.0513279    .0132491
          L3. |  -.0005367   .0162317    -0.03   0.974    -.0323503    .0312768
          L4. |   .0270265   .0158949     1.70   0.089    -.0041269    .0581799
          L5. |  -.0176457   .0161852    -1.09   0.276    -.0493682    .0140767
          L6. |  -.0017395   .0168184    -0.10   0.918     -.034703    .0312239
          L7. |   -.003062   .0154117    -0.20   0.843    -.0332684    .0271445
          L8. |  -.0003618   .0148068    -0.02   0.981    -.0293827     .028659
              |
           q1 |  -.3934526    .477041    -0.82   0.409    -1.328436    .5415305
           q2 |  -.3562299   .4577915    -0.78   0.436    -1.253485    .5410249
           q3 |  -.5884998   .4616218    -1.27   0.202    -1.493262    .3162624
        covid |  -5.703467   .7337732    -7.77   0.000    -7.141636   -4.265298
        _cons |    10.8049   1.247584     8.66   0.000     8.359679    13.25012
--------------+----------------------------------------------------------------
ntlg          |
            g |
          L1. |  -.5207465   1.191368    -0.44   0.662    -2.855784    1.814291
          L2. |  -1.652415   1.189379    -1.39   0.165    -3.983555    .6787258
          L3. |   2.255821   1.234594     1.83   0.068    -.1639379     4.67558
          L4. |  -3.561734   1.184718    -3.01   0.003    -5.883738   -1.239729
          L5. |    1.68842   1.137398     1.48   0.138    -.5408381    3.917679
          L6. |  -3.322423   1.239108    -2.68   0.007    -5.751031   -.8938156
          L7. |   1.585415   1.332996     1.19   0.234    -1.027208    4.198038
          L8. |  -2.189802   1.046499    -2.09   0.036    -4.240903   -.1387009
              |
        ntlg |
          L1. |    .593417   .1369715     4.33   0.000     .3249577    .8618762
          L2. |   .1175421   .1390374     0.85   0.398    -.1549662    .3900504
          L3. |    .022333   .1369923     0.16   0.870    -.2461669    .2908329
          L4. |  -.7311457   .1341494    -5.45   0.000    -.9940737   -.4682177
          L5. |   .3080725      .1366     2.26   0.024     .0403413    .5758037
          L6. |   .1116778   .1419437     0.79   0.431    -.1665267    .3898823
          L7. |   .2732761   .1300718     2.10   0.036       .01834    .5282122
          L8. |  -.4813924   .1249664    -3.85   0.000    -.7263221   -.2364627
              |
           q1 |  -.0602049   4.026128    -0.01   0.988    -7.951271    7.830861
           q2 |  -7.169945   3.863667    -1.86   0.063    -14.74259     .402703
           q3 |  -5.742374   3.895994    -1.47   0.141    -13.37838    1.893634
        covid |  -17.06714   6.192896    -2.76   0.006    -29.20499   -4.929284
        _cons |   37.55826   10.52935     3.57   0.000     16.92111    58.19542
------------------------------------------------------------------------------
```

```
. drop r_g

. predict r_g
(option xb assumed; fitted values)
(8 missing values generated)

. tsline g r_g, xtitle("") title("(e) VAR+Q+C") lcolor("#845B24" "#EEC051") lpa
> ttern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2 "Pre
> dicted")) lwidth(0.6 0.6)

. graph save g5.gph, replace
file g5.gph saved

. var g ntlg, lags(1/8) exog(q1 q2 q3 scar)

Vector autoregression

Sample: 2016q1 thru 2024q4                      Number of obs     =         36
Log likelihood =  -186.7339                     AIC               =   12.70744
FPE            =   1585.023                     HQIC              =   13.35225
Det(Sigma_ml)  =   109.7661                     SBIC              =   14.55488

Equation           Parms      RMSE     R-sq      chi2     P>chi2
------------------------------------------------------------------
g                     21     2.21689   0.7033   85.33334   0.0000
ntlg                  21     13.3646   0.7479   106.7755   0.0000
------------------------------------------------------------------

------------------------------------------------------------------------------
             | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
g            |
           g |
         L1. |   .7328852   .1801209     4.07   0.000     .3798548    1.085916
         L2. |   .0283356   .2139194     0.13   0.895    -.3909388      .44761
         L3. |  -.2070082   .2227629    -0.93   0.353    -.6436155    .2295991
         L4. |  -.5980777   .2171936    -2.75   0.006    -1.023769   -.1723861
         L5. |   .5493809   .1987975     2.76   0.006     .1597449    .9390169
         L6. |  -.2402603   .2177175    -1.10   0.270    -.6669787    .1864582
         L7. |  -.1377798   .2293217    -0.60   0.548    -.5872421    .3116826
         L8. |  -.1226894   .1882073    -0.65   0.514    -.4915689      .24619
             |
        ntlg |
         L1. |     .00593    .024895     0.24   0.812    -.0428632    .0547233
         L2. |  -.0080181   .0252437    -0.32   0.751    -.0574948    .0414587
         L3. |    .027813   .0245834     1.13   0.258    -.0203695    .0759954
         L4. |    .034373   .0256146     1.34   0.180    -.0158307    .0845767
         L5. |  -.0330882   .0246472    -1.34   0.179    -.0813957    .0152194
```

```
         L6. |   .0212037   .0254006     0.83   0.404    -.0285805    .0709879
         L7. |   .0087137   .0245323     0.36   0.722    -.0393687    .0567962
         L8. |   .0192792   .0232922     0.83   0.408    -.0263726    .0649311
             |
          q1 |  -.5557772   .7271065    -0.76   0.445     -1.98088    .8693254
          q2 |  -.2117284   .6979747    -0.30   0.762    -1.579734    1.156277
          q3 |  -.8140489   .7033983    -1.16   0.247    -2.192684    .5645865
        scar |   -1.96991   .8472875    -2.32   0.020    -3.630563   -.3092573
       _cons |   5.054532   1.513563     3.34   0.001     2.088003    8.02106
-------------+----------------------------------------------------------------
ntlg         |
           g |
         L1. |   1.927133   1.085867     1.77   0.076    -.2011264    4.055393
         L2. |  -1.971064   1.289623    -1.53   0.126    -4.498678    .5565502
         L3. |   2.497362   1.342936     1.86   0.063    -.1347446    5.129468
         L4. |  -2.270668   1.309361    -1.73   0.083    -4.836968    .2956331
         L5. |   2.797295    1.19846     2.33   0.020     .4483568    5.146232
         L6. |  -4.127351   1.312519    -3.14   0.002    -6.699842    -1.55486
         L7. |    2.91911   1.382476     2.11   0.035     .2095069    5.628713
         L8. |  -1.188529   1.134616    -1.05   0.295    -3.412335    1.035277
             |
        ntlg |
         L1. |   .4590193   .1500804     3.06   0.002     .1648671    .7531715
         L2. |   .1204277   .1521829     0.79   0.429    -.1778453    .4187007
         L3. |   .0719235   .1482018     0.49   0.627    -.2185467    .3623937
         L4. |  -.8121567   .1544186    -5.26   0.000    -1.114812   -.5095017
         L5. |   .2003453   .1485865     1.35   0.178     -.090879    .4915695
         L6. |   .1645056   .1531284     1.07   0.283    -.1356206    .4646318
         L7. |   .2242071   .1478941     1.52   0.130    -.0656601    .5140743
         L8. |  -.4919108   .1404179    -3.50   0.000    -.7671247   -.2166968
             |
          q1 |   -.667193   4.383393    -0.15   0.879    -9.258486      7.9241
          q2 |  -6.719922   4.207771    -1.60   0.110      -14.967    1.527157
          q3 |  -6.289902   4.240468    -1.48   0.138    -14.60107    2.021262
        scar |    4.25227   5.107909     0.83   0.405    -5.759048    14.26359
       _cons |   5.479746    9.12458     0.60   0.548     -12.4041    23.36359
------------------------------------------------------------------------------

. drop r_g

. predict r_g
(option xb assumed; fitted values)
(8 missing values generated)

. tsline g r_g, xtitle("") title("(f) VAR+Q+S") lcolor("#845B24" "#EEC051") lpa
> ttern(solid shortdash) xlabel(, format(%ty)) legend(order(1 "Observed" 2 "Pre
> dicted")) lwidth(0.6 0.6)
```

```
. graph save g6.gph, replace
file g6.gph saved

.
. graph combine g1.gph g2.gph g3.gph g4.gph g5.gph g6.gph, rows(3) cols(2) imar
> gin(2 2 2 2)

. graph export "fig/VARstata.png", width(2400) height(3200) replace
file fig/VARstata.png written in PNG format

.
```



0.3.8 ARDL with growth dataset

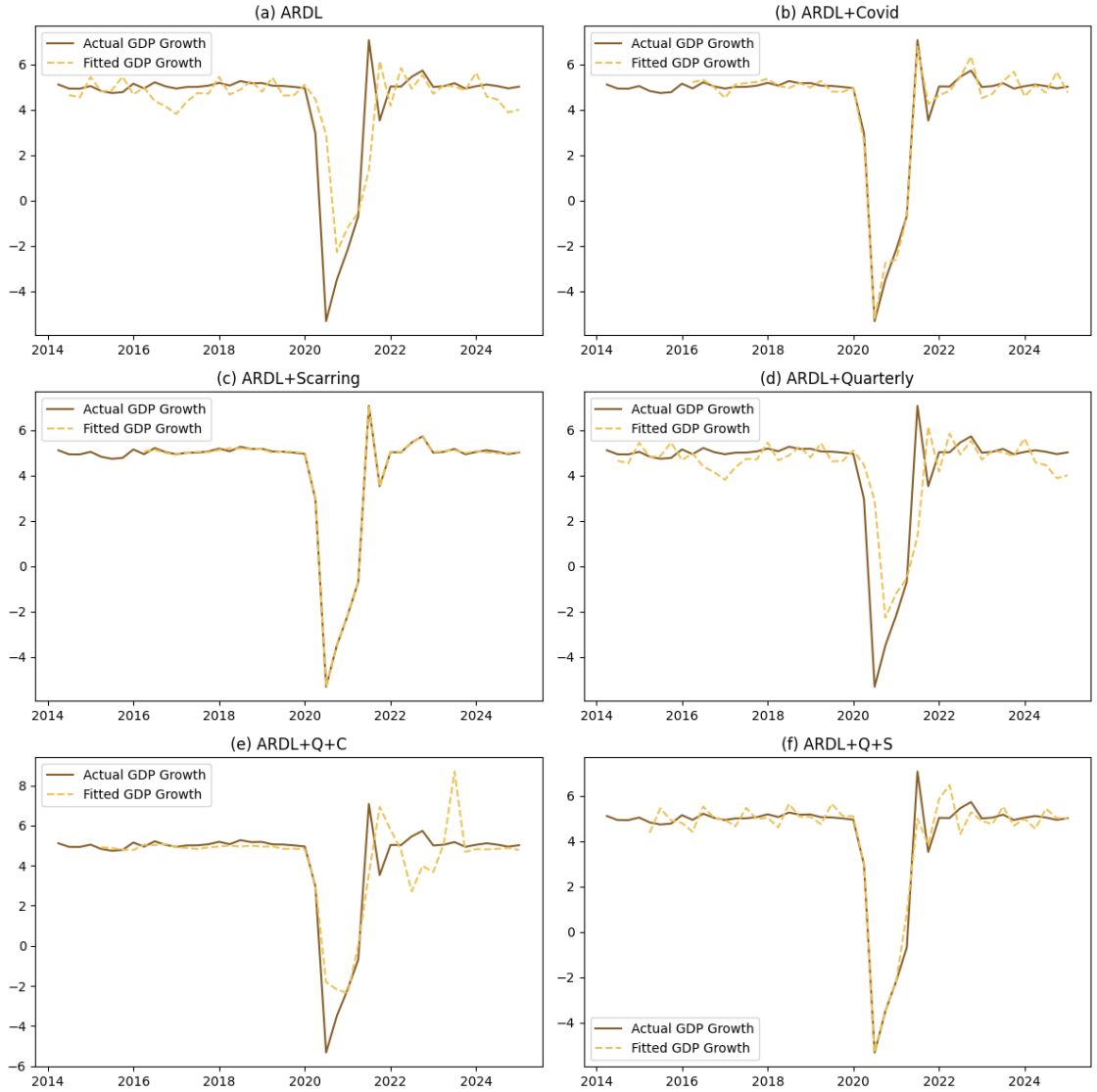
[12]:
```
## ARDL with growth

en=ntl[['g']]
ex=ntl[['ntlg']]
exc=ntl[['ntlg','covid']]
exs=ntl[['ntlg','scar']]
exq=ntl[['ntlg','q1','q2','q3']]
exqc=ntl[['ntlg','q1','q2','q3','covid']]
exqs=ntl[['ntlg','q1','q2','q3','scar']]
```

```
lags = ardl_select_order(endog=en, exog=ex, maxlag=8,maxorder=8,␣
 ↪ic='aic',seasonal=False)
ve = ARDL(endog=en,lags=lags.ar_lags,exog=ex,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exc, maxlag=8,maxorder=8,␣
 ↪ic='aic',seasonal=False)
vec= ARDL(endog=en,lags=lags.ar_lags,exog=exc,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exs, maxlag=8,maxorder=8,␣
 ↪ic='aic',seasonal=False)
ves= ARDL(endog=en,lags=lags.ar_lags,exog=exs,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exq, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
veq= ARDL(endog=en,lags=lags.ar_lags,exog=exq,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exqc, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
veqc= ARDL(endog=en,lags=lags.ar_lags,exog=exqc,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exqs, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
veqs= ARDL(endog=en,lags=lags.ar_lags,exog=exqs,order=lags.dl_lags,trend='ct').
 ↪fit()

models = {'fve': ve, 'fvec': vec,'fves': ves}
results = {}

for name, model in models.items():
    fitted = pd.DataFrame(model.predict(), columns=en.columns)
    fitted.index = pd.date_range(end='2024-12-31', periods=44, freq='QE')
    merged = pd.merge(en, fitted, left_index=True, right_index=True,␣
 ↪suffixes=('', f'_fitted'))
    results[name] = merged

models = {'fveq': veq,'fveqc': veqc, 'fveqs': veqs}
result = {}

for name, model in models.items():
    fitted = pd.DataFrame(model.predict(), columns=en.columns)
    fitted.index = pd.date_range(end='2024-12-31', periods=44, freq='QE')
    merged = pd.merge(en, fitted, left_index=True, right_index=True,␣
 ↪suffixes=('', f'_fitted'))
    result[name] = merged
```

```
fig, ax = plt.subplots(3,2,figsize=(12, 12))

ax[0,0].plot(results['fve']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,0].plot(results['fve']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[0,0].set_title('(a) ARDL')
ax[0,0].legend()

ax[0,1].plot(results['fvec']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,1].plot(results['fvec']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[0,1].set_title('(b) ARDL+Covid')
ax[0,1].legend()

ax[1,0].plot(results['fves']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,0].plot(results['fves']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,0].set_title('(c) ARDL+Scarring')
ax[1,0].legend()

ax[1,1].plot(result['fveq']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,1].plot(result['fveq']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,1].set_title('(d) ARDL+Quarterly')
ax[1,1].legend()

ax[2,0].plot(result['fveqc']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,0].plot(result['fveqc']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,0].set_title('(e) ARDL+Q+C')
ax[2,0].legend()

ax[2,1].plot(result['fveqs']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,1].plot(result['fveqs']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,1].set_title('(f) ARDL+Q+S')
ax[2,1].legend()
plt.tight_layout()
plt.savefig("fig/ARDL.png")
```

```
plt.show()
```

```
c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\ardl\model.py:455: SpecificationWarning: exog contains
variables that are missing from the order dictionary.  Missing keys: q2, q3, q1.
  return _format_order(self.data.orig_exog, order, self._causal)
c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\ardl\model.py:455: SpecificationWarning: exog contains
variables that are missing from the order dictionary.  Missing keys: ntlg, q2,
q1, q3.
  return _format_order(self.data.orig_exog, order, self._causal)
c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\ardl\model.py:455: SpecificationWarning: exog contains
variables that are missing from the order dictionary.  Missing keys: ntlg, q3.
  return _format_order(self.data.orig_exog, order, self._causal)
```



[13]:
```
print(ves.summary())
```

ARDL Model Results

Dep. Variable:	g	No. Observations:	44
Model:	ARDL(7, 8, 5)	Log Likelihood	61.267
Method:	Conditional MLE	S.D. of innovations	0.046
Date:	Wed, 01 Oct 2025	AIC	-72.534
Time:	22:03:15	BIC	-32.261
Sample:	03-31-2016 - 12-31-2024	HQIC	-58.336

	coef	std err	z	P>|z|	[0.025	0.975]
const	7.0042	0.122	57.419	0.000	6.741	7.268
trend	-0.0058	0.005	-1.242	0.236	-0.016	0.004
g.L1	-0.3732	0.022	-17.270	0.000	-0.420	-0.326
g.L2	-0.0345	0.020	-1.696	0.114	-0.078	0.009
g.L3	-0.0170	0.018	-0.923	0.373	-0.057	0.023
g.L4	0.0309	0.015	2.042	0.062	-0.002	0.064
g.L5	0.1009	0.014	7.210	0.000	0.071	0.131
g.L6	-0.0235	0.016	-1.464	0.167	-0.058	0.011
g.L7	-0.0387	0.014	-2.847	0.014	-0.068	-0.009
ntlg.L0	0.0004	0.002	0.188	0.854	-0.004	0.004
ntlg.L1	0.0027	0.002	1.682	0.116	-0.001	0.006
ntlg.L2	-0.0009	0.001	-0.655	0.524	-0.004	0.002
ntlg.L3	0.0008	0.001	0.576	0.574	-0.002	0.004
ntlg.L4	0.0007	0.002	0.327	0.749	-0.004	0.006
ntlg.L5	-0.0020	0.001	-1.372	0.193	-0.005	0.001
ntlg.L6	-0.0028	0.001	-1.945	0.074	-0.006	0.000
ntlg.L7	-0.0009	0.001	-0.641	0.533	-0.004	0.002
ntlg.L8	-0.0027	0.002	-1.622	0.129	-0.006	0.001
scar.L0	-2.1390	0.099	-21.515	0.000	-2.354	-1.924
scar.L1	-8.8914	0.147	-60.691	0.000	-9.208	-8.575
scar.L2	-1.2987	0.209	-6.205	0.000	-1.751	-0.847
scar.L3	1.6116	0.197	8.178	0.000	1.186	2.037
scar.L4	2.0640	0.212	9.715	0.000	1.605	2.523
scar.L5	8.8103	0.198	44.540	0.000	8.383	9.238

0.3.9 OLS Regression

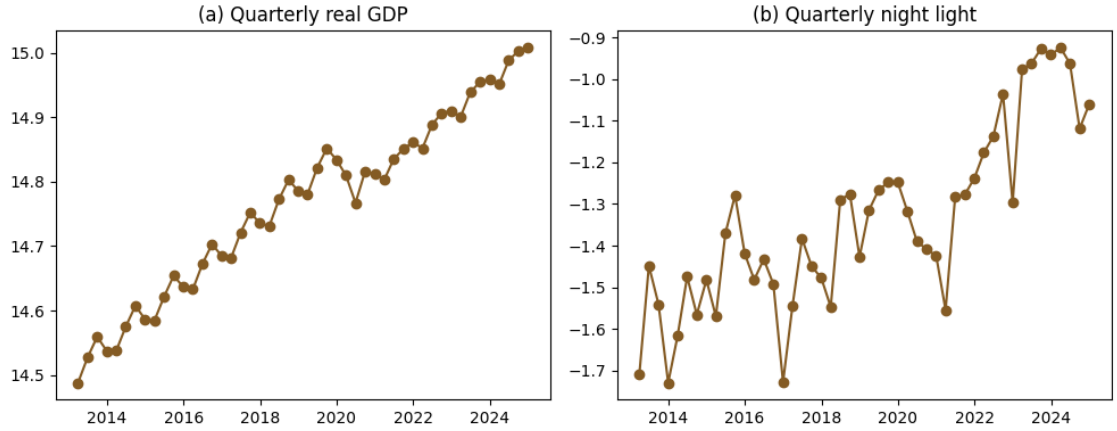
0.4 Quarterly Real GDP vs Quarterly NTL.

0.4.1 Dataset

turn on the last line to see the dataframe.

[14]:
```
## Data prep
### Creating data
ntl=pd.read_csv('ntl_monthly_avg_2012-2024.csv')
gdp=pd.read_excel('GDP_YoY_Quarterly_12_24.xlsx')

### Make time index
ntl.Date=pd.to_datetime(ntl['Date'])
ntl['qtr']=ntl['Date'].dt.quarter
ntl['year']=ntl['Date'].dt.year
### Averaging the radiance into quarterly, make it yoy quarterly growth
ntl=ntl.groupby(['year','qtr'])['NTL_Radiance'].mean().reset_index()
ntl['Date']=pd.date_range(start='2012-01-01', periods=len(ntl), freq='QE')
ntl=ntl[['Date','NTL_Radiance']]
ntl['g']=np.log(gdp['GDP'])
ntl['ntlg']=np.log(ntl['NTL_Radiance'])
#ntl['NTL_Radiancelag'] = ntl['NTL_Radiance'].shift(4)
#ntl['ntlg'] = ((ntl['NTL_Radiance'] - ntl['NTL_Radiancelag']) /␣
 ↪ntl['NTL_Radiancelag']) * 100
### Creating dummy quarterly and dummy covid
ntl['q1']=np.where(ntl['Date'].dt.quarter==1,1,0)
ntl['q2']=np.where(ntl['Date'].dt.quarter==2,1,0)
ntl['q3']=np.where(ntl['Date'].dt.quarter==3,1,0)
ntl['q4']=np.where(ntl['Date'].dt.quarter==4,1,0)
ntl['covid']=np.where((ntl['Date'].dt.year>=2020) & (ntl['Date'].dt.
 ↪year<=2022),1,0)
ntl['scar']=np.where((ntl['Date'].dt.year>=2020) ,1,0)
### Back to making time index
ntl=ntl.dropna().reset_index(drop=True)
ntl=ntl.set_index('Date')
ntl=ntl.asfreq('QE-DEC')
#ntlm=ntlm[['g','ntlg']]

### Creating dummy quarterly and dummy covid

## OLS-ing
mod=sm.OLS(ntl['g'], sm.add_constant(ntl['ntlg'])).fit()
ntl['resid']=mod.resid
ntl['ols']=mod.predict()
#ntl
```

[15]:
```
# Plotting GDP Growth and Night light growth side by side

fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))
ntlm=ntl[4:]
ax1.plot(ntlm['g'],color='#845B24',marker='o', linestyle='-')
ax1.set_title('(a) Quarterly real GDP')

ax2.plot(ntlm['ntlg'], linestyle='-', color='#845B24',marker='o')
ax2.set_title('(b) Quarterly night light')

plt.tight_layout()
plt.savefig("fig/figQ.png") # Turn off to not save, or change file name to save␣
 ↪in your preferred location
plt.show()
```



0.4.2 OLS and residuals

[16]:
```
## OLS results and plotting residuals
ntl=ntlm
print(mod.summary())
fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(10, 4))

ax1.plot(ntlm['g'],color='#845B24',linestyle="-",label="observed GDP Growth")
ax1.plot(ntlm['ols'],color='#EEC051',linestyle="--",label="OLS-fitted GDP␣
 ↪Growth")
ax1.set_title('(a) GDP Growth')
ax1.legend()

ax2.plot(ntlm['resid'], linestyle='-', color='#845B24')
ax2.set_title('(b) OLS Residuals')
```

```
plt.tight_layout()
plt.savefig("fig/Qols.png") # Turn off to not save, or change file name to save␣
  ↪in your preferred location
plt.show()
```

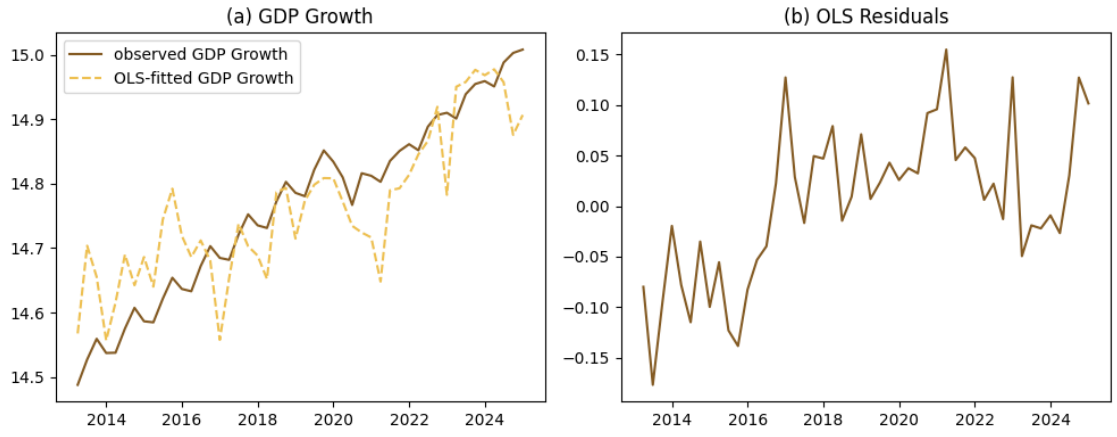
OLS Regression Results

Dep. Variable:	g	R-squared:	0.737
Model:	OLS	Adj. R-squared:	0.731
Method:	Least Squares	F-statistic:	139.9
Date:	Wed, 01 Oct 2025	Prob (F-statistic):	4.20e-16
Time:	22:03:16	Log-Likelihood:	57.804
No. Observations:	52	AIC:	-111.6
Df Residuals:	50	BIC:	-107.7
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	15.4614	0.062	250.049	0.000	15.337	15.586
ntlg	0.5232	0.044	11.829	0.000	0.434	0.612

Omnibus:	0.458	Durbin-Watson:	0.829
Prob(Omnibus):	0.795	Jarque-Bera (JB):	0.591
Skew:	0.000	Prob(JB):	0.744
Kurtosis:	2.478	Cond. No.	11.5

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.



0.4.3 ADF test of the series and residuals.

We don't first diff in this step cuz we do another ADF test on the growth dataset.

[17]:
```
## ADF Test for g, ntlg and OLS residuals
def adf_test(series, name=""):
    """
    Perform ADF test and print results
    """
    result = adfuller(series.dropna(), autolag="BIC")
    print(f"ADF Test for {name}")
    print(f"  Test Statistic : {result[0]:.4f}")
    print(f"  p-value        : {result[1]:.4f}")
    print(f"  #Lags Used     : {result[2]}")
    print(f"  #Observations  : {result[3]}")
    for key, value in result[4].items():
        print(f"   Critical Value {key} : {value:.4f}")
    if result[1] <= 0.05:
        print(f"  ==> {name} is stationary\n (reject H0 of unit root)\n")
    else:
        print(f"  ==> {name} is non-stationary\n (fail to reject H0)\n")

# Run ADF tests for both series
adf_test(ntlm["g"], "GDP YoY Growth")
adf_test(ntlm["ntlg"], "NTL YoY Growth")
adf_test(ntlm["resid"], "OLS Residuals")
```

```
ADF Test for GDP YoY Growth
  Test Statistic : -0.4807
  p-value        : 0.8957
  #Lags Used     : 4
  #Observations  : 43
   Critical Value 1% : -3.5925
   Critical Value 5% : -2.9315
   Critical Value 10% : -2.6041
  ==> GDP YoY Growth is non-stationary
 (fail to reject H0)

ADF Test for NTL YoY Growth
  Test Statistic : -2.3235
  p-value        : 0.1645
  #Lags Used     : 0
  #Observations  : 47
   Critical Value 1% : -3.5778
   Critical Value 5% : -2.9253
   Critical Value 10% : -2.6008
  ==> NTL YoY Growth is non-stationary
 (fail to reject H0)
```

```
ADF Test for OLS Residuals
  Test Statistic : -3.1182
  p-value        : 0.0252
  #Lags Used     : 0
  #Observations  : 47
   Critical Value 1% : -3.5778
   Critical Value 5% : -2.9253
   Critical Value 10% : -2.6008
  ==> OLS Residuals is stationary
 (reject H0 of unit root)
```

0.4.4 Johansen cointegration test

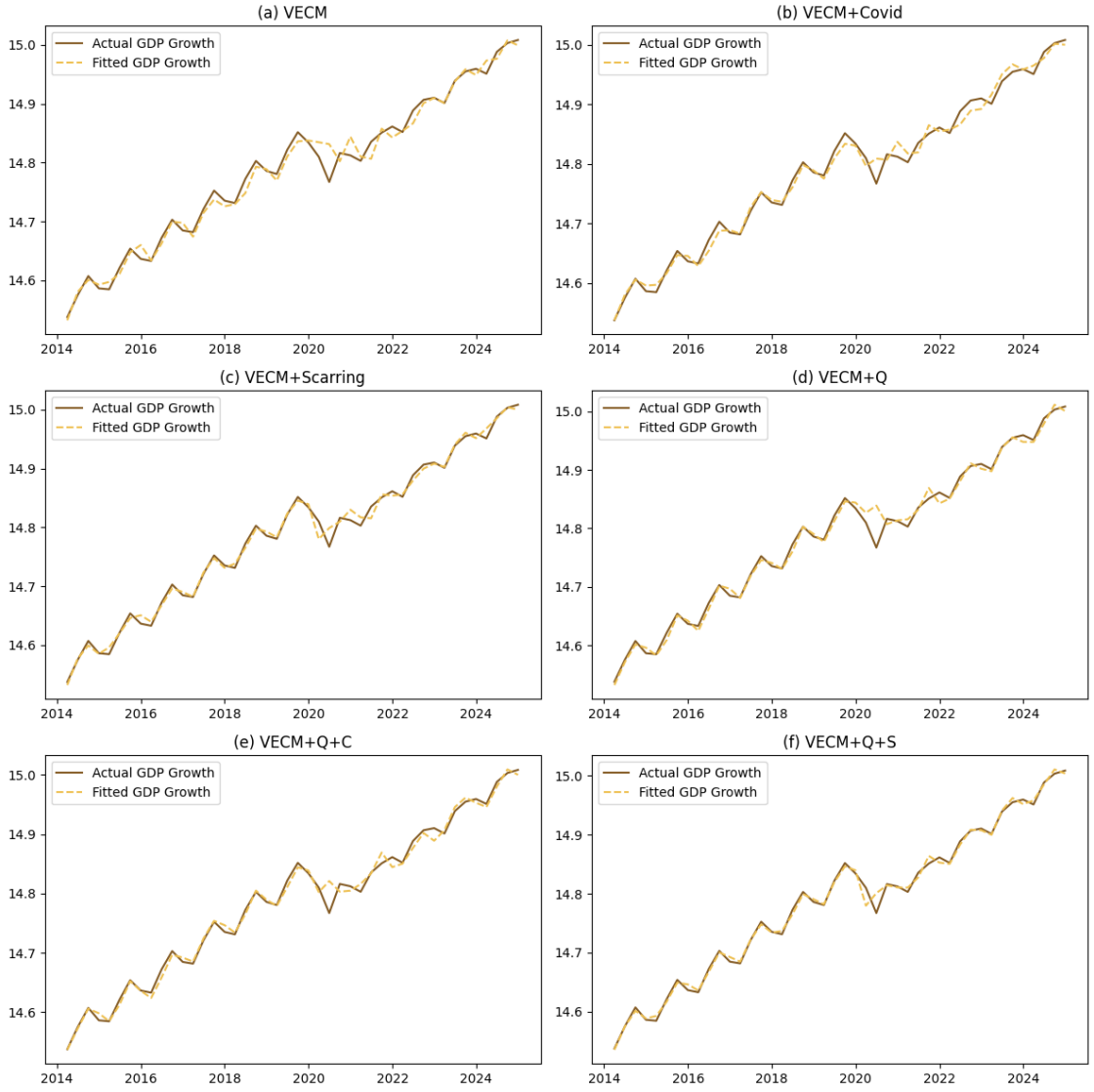
[18]:
```
# Select optimal lag order
ntlm=ntl[['g','ntlg']]
lag_order = select_order(ntlm, maxlags=12, deterministic="ci")
print(lag_order.summary())

# Select cointegration rank
coint_rank = select_coint_rank(ntlm, det_order=0, k_ar_diff=lag_order.bic)
print(coint_rank.summary())
```

```
 VECM Order Selection (* highlights the minimums)
=================================================
       AIC         BIC          FPE         HQIC
-------------------------------------------------
0       -11.94     -11.58*   6.547e-06      -11.82
1       -11.86      -11.33   7.124e-06      -11.67
2       -12.00      -11.29   6.265e-06      -11.75
3      -12.42*      -11.53  4.188e-06*     -12.11*
4       -12.27      -11.20   4.989e-06      -11.90
5       -12.12      -10.88   5.971e-06      -11.69
6       -12.11      -10.69   6.377e-06      -11.62
7       -12.18      -10.58   6.353e-06      -11.63
8       -12.19      -10.42   6.919e-06      -11.58
9       -12.27      -10.31   7.310e-06      -11.59
10      -12.22      -10.09   9.146e-06      -11.48
11      -12.09      -9.782   1.318e-05      -11.30
12      -12.34      -9.849   1.448e-05      -11.48
-------------------------------------------------
Johansen cointegration test using trace test statistic with 5% significance
level
=====================================
r_0 r_1 test statistic critical value
-------------------------------------
  0   2          19.26          15.49
  1   2         0.1092          3.841
```

0.4.5 VECM with quarterly dataset

[19]:
```
en=ntl[['g','ntlg']]
exc=ntl[['covid']]
exs=ntl[['scar']]
exq=ntl[['q1','q2','q3']]
exqc=ntl[['q1','q2','q3','covid']]
exqs=ntl[['q1','q2','q3','scar']]

lag=lag_order.aic

ve = VECM(en,k_ar_diff=lag, coint_rank=1, deterministic="cili").fit()
vec = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exc,deterministic="cili").fit()
ves = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exs,deterministic="cili").fit()
veq = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exq,deterministic="cili").fit()
veqc = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exqc,deterministic="cili").
 ↪fit()
veqs = VECM(en,k_ar_diff=lag, coint_rank=1, exog=exqs,deterministic="cili").
 ↪fit()

models = {'fve': ve, 'fvec': vec,'fves': ves, 'fveq': veq,'fveqc': veqc,␣
 ↪'fveqs': veqs}
results = {}

for name, model in models.items():
    fitted = pd.DataFrame(model.fittedvalues, columns=en.columns)
    fitted.index = pd.date_range(end='2024-12-31', periods=44, freq='QE')
    merged = pd.merge(en, fitted, left_index=True, right_index=True,␣
 ↪suffixes=('', f'_fitted'))
    results[name] = merged


fig, ax = plt.subplots(3,2,figsize=(12, 12))

ax[0,0].plot(results['fve']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,0].plot(results['fve']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[0,0].set_title('(a) VECM')
ax[0,0].legend()

ax[0,1].plot(results['fvec']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,1].plot(results['fvec']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
```

```
ax[0,1].set_title('(b) VECM+Covid')
ax[0,1].legend()

ax[1,0].plot(results['fves']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,0].plot(results['fves']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,0].set_title('(c) VECM+Scarring')
ax[1,0].legend()

ax[1,1].plot(results['fveq']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,1].plot(results['fveq']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,1].set_title('(d) VECM+Q')
ax[1,1].legend()

ax[2,0].plot(results['fveqc']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,0].plot(results['fveqc']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,0].set_title('(e) VECM+Q+C')
ax[2,0].legend()

ax[2,1].plot(results['fveqs']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,1].plot(results['fveqs']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,1].set_title('(f) VECM+Q+S')
ax[2,1].legend()
plt.tight_layout()
plt.savefig("fig/VECMQ.png")
plt.show()
```



[20]:
```
print(veqs.summary())
```

Det. terms outside the coint. relation & lagged endog. parameters for equation g

	coef	std err	z	P>|z|	[0.025	0.975]
exog1	-0.0101	0.005	-1.860	0.063	-0.021	0.001
exog2	0.0065	0.006	1.120	0.263	-0.005	0.018
exog3	0.0180	0.004	4.128	0.000	0.009	0.026
exog4	-0.0596	0.007	-7.979	0.000	-0.074	-0.045
L1.g	-0.1187	0.079	-1.505	0.132	-0.273	0.036
L1.ntlg	0.0075	0.014	0.546	0.585	-0.019	0.034
L2.g	-0.0404	0.088	-0.461	0.645	-0.212	0.131

```
L2.ntlg       -0.0192      0.016     -1.165      0.244      -0.052       0.013
L3.g          -0.0434      0.107     -0.407      0.684      -0.252       0.166
L3.ntlg       -0.0229      0.015     -1.500      0.134      -0.053       0.007
Det. terms outside the coint. relation & lagged endog. parameters for equation
ntlg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
exog1         -0.0020      0.057     -0.035      0.972      -0.114       0.110
exog2         -0.0041      0.061     -0.068      0.946      -0.124       0.115
exog3         -0.0214      0.046     -0.466      0.642      -0.111       0.069
exog4         -0.1394      0.079     -1.770      0.077      -0.294       0.015
L1.g           1.1129      0.832      1.338      0.181      -0.517       2.743
L1.ntlg       -0.3544      0.145     -2.448      0.014      -0.638      -0.071
L2.g          -1.8351      0.925     -1.985      0.047      -3.647      -0.023
L2.ntlg       -0.0595      0.174     -0.342      0.732      -0.400       0.282
L3.g           1.7605      1.124      1.566      0.117      -0.443       3.964
L3.ntlg        0.0294      0.161      0.183      0.855      -0.286       0.345
                Loading coefficients (alpha) for equation g
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ec1           -0.7240      0.092     -7.850      0.000      -0.905      -0.543
              Loading coefficients (alpha) for equation ntlg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ec1           -1.6220      0.972     -1.668      0.095      -3.528       0.284
          Cointegration relations for loading-coefficients-column 1
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
beta.1         1.0000          0          0      0.000       1.000       1.000
beta.2        -0.0231      0.022     -1.070      0.285      -0.065       0.019
const        -14.5487      0.038   -385.056      0.000     -14.623     -14.475
lin_trend     -0.0121      0.000    -27.535      0.000      -0.013      -0.011
==============================================================================
```

0.5 ARDL with quarterly dataset

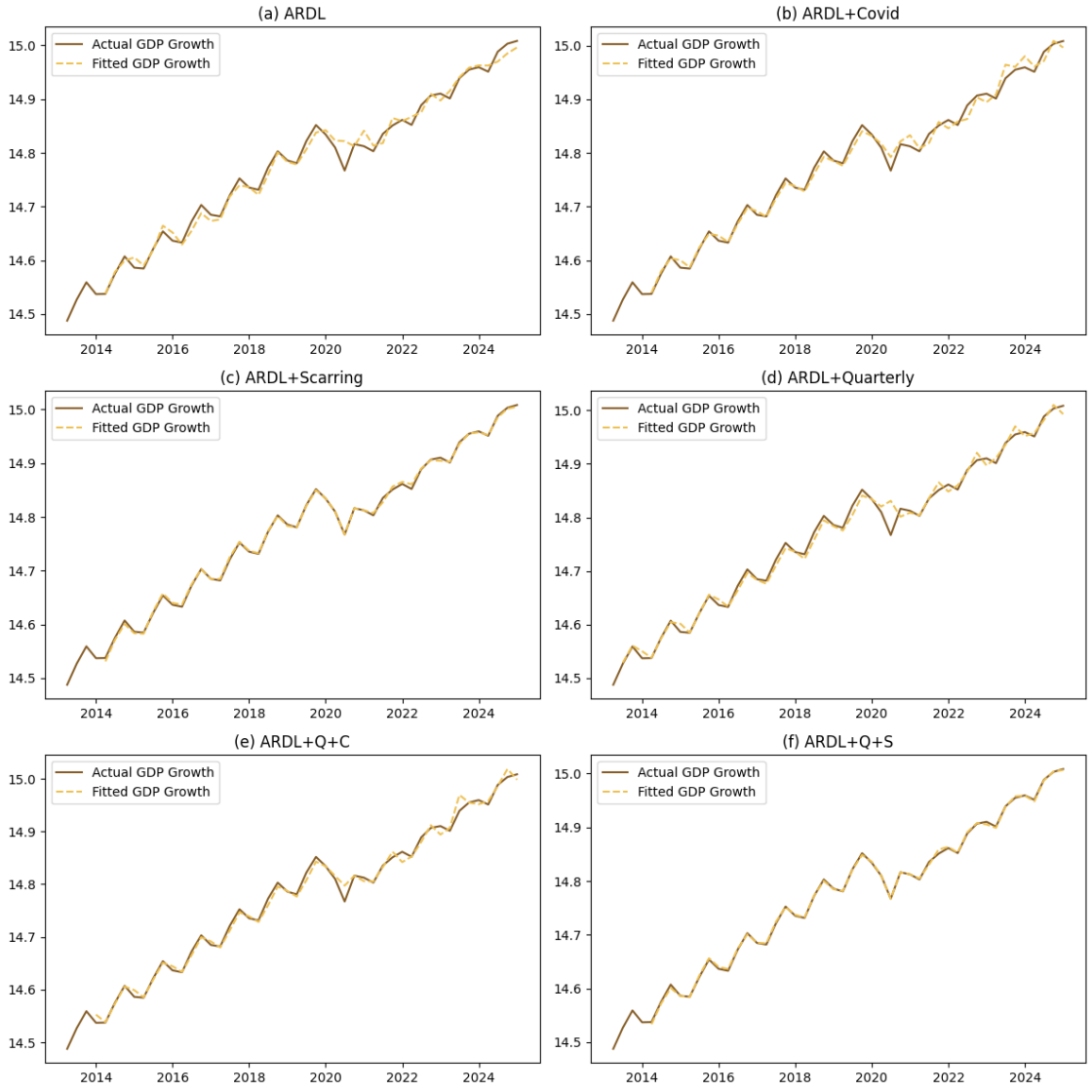
[21]:
```
en=ntl[['g']]
ex=ntl[['ntlg']]
exc=ntl[['ntlg','covid']]
exs=ntl[['ntlg','scar']]
exq=ntl[['ntlg','q1','q2','q3']]
exqc=ntl[['ntlg','q1','q2','q3','covid']]
exqs=ntl[['ntlg','q1','q2','q3','scar']]
```

```
lags = ardl_select_order(endog=en, exog=ex, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
ve = ARDL(endog=en,lags=lags.ar_lags,exog=ex,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exc, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
vec= ARDL(endog=en,lags=lags.ar_lags,exog=exc,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exs, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
ves= ARDL(endog=en,lags=lags.ar_lags,exog=exs,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exq, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
veq= ARDL(endog=en,lags=lags.ar_lags,exog=exq,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exqc, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
veqc= ARDL(endog=en,lags=lags.ar_lags,exog=exqc,order=lags.dl_lags,trend='ct').
 ↪fit()
lags = ardl_select_order(endog=en, exog=exqs, maxlag=4,maxorder=4,␣
 ↪ic='aic',seasonal=False)
veqs= ARDL(endog=en,lags=lags.ar_lags,exog=exqs,order=lags.dl_lags,trend='ct').
 ↪fit() # This looks too good to be true

models = {'fve': ve, 'fvec': vec,'fves': ves, 'fveq': veq,'fveqc': veqc,␣
 ↪'fveqs': veqs}
results = {}

for name, model in models.items():
    fitted = pd.DataFrame(model.predict(), columns=en.columns)
    fitted.index = pd.date_range(end='2024-12-31', periods=48, freq='QE')
    merged = pd.merge(en, fitted, left_index=True, right_index=True,␣
 ↪suffixes=('', f'_fitted'))
    results[name] = merged


fig, ax = plt.subplots(3,2,figsize=(12, 12))

ax[0,0].plot(results['fve']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,0].plot(results['fve']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[0,0].set_title('(a) ARDL')
ax[0,0].legend()
```

```
ax[0,1].plot(results['fvec']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[0,1].plot(results['fvec']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[0,1].set_title('(b) ARDL+Covid')
ax[0,1].legend()

ax[1,0].plot(results['fves']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,0].plot(results['fves']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,0].set_title('(c) ARDL+Scarring')
ax[1,0].legend()

ax[1,1].plot(results['fveq']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[1,1].plot(results['fveq']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[1,1].set_title('(d) ARDL+Quarterly')
ax[1,1].legend()

ax[2,0].plot(results['fveqc']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,0].plot(results['fveqc']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,0].set_title('(e) ARDL+Q+C')
ax[2,0].legend()

ax[2,1].plot(results['fveqs']['g'],color='#845B24',linestyle='-',label="Actual␣
 ↪GDP Growth")
ax[2,1].plot(results['fveqs']['g_fitted'], linestyle='--', color='#EEC051',␣
 ↪label="Fitted GDP Growth")
ax[2,1].set_title('(f) ARDL+Q+S')
ax[2,1].legend()
plt.tight_layout()
plt.savefig("fig/ARDLQ.png") # Turn off to not save, or change file name to␣
 ↪save in your preferred location
plt.show()
```

```
c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\ardl\model.py:455: SpecificationWarning: exog contains
variables that are missing from the order dictionary.  Missing keys: ntlg.
  return _format_order(self.data.orig_exog, order, self._causal)
c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\ardl\model.py:455: SpecificationWarning: exog contains
variables that are missing from the order dictionary.  Missing keys: q1.
```

```
  return _format_order(self.data.orig_exog, order, self._causal)
c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\ardl\model.py:455: SpecificationWarning: exog contains
variables that are missing from the order dictionary.  Missing keys: ntlg, q1.
  return _format_order(self.data.orig_exog, order, self._causal)
c:\Users\imed\AppData\Local\Programs\Python\Python313\Lib\site-
packages\statsmodels\tsa\ardl\model.py:455: SpecificationWarning: exog contains
variables that are missing from the order dictionary.  Missing keys: q2.
  return _format_order(self.data.orig_exog, order, self._causal)
```



[22]:
```
print(veqs.summary())
```

```
                              ARDL Model Results
==============================================================================
```

Dep. Variable:	g	No. Observations:	48
Model:	ARDL(4, 0, 0, 0, 4)	Log Likelihood	199.333
Method:	Conditional MLE	S.D. of innovations	0.003
Date:	Wed, 01 Oct 2025	AIC	-368.665
Time:	22:03:23	BIC	-341.902
Sample:	03-31-2014 - 12-31-2024	HQIC	-358.740

	coef	std err	z	P>|z|	[0.025	0.975]
const	39.5332	5.680	6.960	0.000	27.932	51.134
trend	0.0335	0.005	6.926	0.000	0.024	0.043
g.L1	-0.6836	0.122	-5.583	0.000	-0.934	-0.434
g.L2	-0.4758	0.139	-3.423	0.002	-0.760	-0.192
g.L3	-0.5943	0.128	-4.633	0.000	-0.856	-0.332
g.L4	0.0250	0.068	0.366	0.717	-0.115	0.165
ntlg.L0	0.0128	0.005	2.495	0.018	0.002	0.023
q1.L0	-0.0110	0.003	-4.274	0.000	-0.016	-0.006
q3.L0	0.0087	0.003	3.125	0.004	0.003	0.014
scar.L0	-0.0210	0.004	-5.829	0.000	-0.028	-0.014
scar.L1	-0.0946	0.005	-17.540	0.000	-0.106	-0.084
scar.L2	-0.0448	0.013	-3.426	0.002	-0.071	-0.018
scar.L3	-0.0293	0.012	-2.347	0.026	-0.055	-0.004
scar.L4	-0.0325	0.011	-3.058	0.005	-0.054	-0.011