

FINANCIAL ECONOMETRICS 1 - M2 FTD

EMPIRICAL APPLICATIONS

Luis Miguel Fonseca

Stéphane Eloundou Mvondo

Natalia Cárdenas Frías

December 13, 2023

Contents

Introduction	3
1 Series Dynamics	3
1.1 Data	3
1.2 Unit root and trends	4
1.2.1 ADF - Test jointly for deterministic and stochastic trend (with drift)	4
1.2.2 ADF - Test jointly for stochastic trend and drift	6
1.2.3 ADF - Test for stochastic trend only	7
1.2.4 Check stationarity of the series in deltas if UR in levels	7
1.3 Decomposition of the series	8
1.3.1 Estimation of the parameters in the stationary series	8
1.4 Cyclical component	12
2 Canonical VAR model application	12
3 Cointegration theory	14
4 Impulse Response Analysis	14
4.1 Canonical IRF	14
4.2 Structural IRF	14
5 Introduce non-linearities	14
5.1 Markov-switching model	14
5.2 STR model	14

Appendix A Code - Data Cleaning	15
Appendix B Code - Analysis 1	20

Introduction

something, probably describe how all applications make sense one after the other and what the research question we could have made ourselves when doing the applications, try to give a coherent look to the whole thing.

This document compiles all our applications for the Financial Econometrics course. Each section represents a specific application, but we tried to make them coherent across them around a broad question:

1 Series Dynamics

1.1 Data

Note: Depending on each exercise along these applications we might use different series. In this first section, we performed the stationarity and component analysis of all of them to be able to use them rapidly without having to worry about seasonality or the presence of UR. Therefore, this section encompasses more than the 3 series that were asked in the exercise.

In this work, we focus on the US market. We use the following series retrieved for the most part from FRED with its Python API (FRED tickers are in square brackets) :

Inflation Expectation [MICH] This data series is made public by the University of Michigan from their Survey of Consumers. The series represents the median expected value of the percent change in prices over the next year. The series is not seasonally adjusted.

GDP deflator [A191RI1Q225SBEA] As a measure of inflation, we decided to use the implicit price deflator of the US GDP. Unlike measures like the CPI deflators do not consider baskets of goods and therefore are broader measures of the price changes across the entire economy that measure the ratio of the GDP in value and volume. It is a measure produced by the US Bureau of Economic Analysis as a quarterly measure of percent change QoQ. The raw series is already seasonally adjusted at an annual rate.

Unemployment rate [UNRATENSA] It represents the share (in percent) of unemployed people over the labor force ¹. The source is the 'Current Population Survey (Household Survey)' of the US Bureau of Labor Statistics

FED fund rate

S&P 500 price

¹The labor force data in this context encompasses people older than 16, living in the continental US, who don't reside in institutions like prisons or homes for the aged, and who are not in active duty in the military.

1.2 Unit root and trends

As for any time series analysis, the first analysis to perform is regarding the presence of unit roots in the series that would make them non-stationary. To do so, we perform the Augmented Dickey-Fuller tests that evaluate the presence of a stochastic trend (a unit root), a deterministic trend, and an intercept or drift. Importantly, this test requires estimating three equations/specifications because it requires investigating the joint presence of both types of trends and drift, for them to discard elements one by one. Each specification tests a different data-generating process of the series. For all specifications, the main null hypothesis H_0 is that the series exhibits a UR. The inference with this test is non-standard and requires to use of corrected critical values to assess significance with the t-statistics.

1.2.1 ADF - Test jointly for deterministic and stochastic trend (with drift)

We first run the following specification to the ADF test to *jointly* investigate the presence of a stochastic and a determinist trend for each series $(X_t)_t$:

$$\Delta X_t = \alpha + \beta t + \gamma X_{t-1} + \sum_{i=1,2,\dots} \rho_i \Delta X_{t-i} + \varepsilon_t \quad (1)$$

As per usual, the ADF test assumes $H_0: \gamma = 0$ i.e. a unit root exists and the series is non-stationary. We use R's built-in function `ur.df` with `type='trend'` to get this estimation. This function gives us (i) a regression table per series and (ii) a summary table with the following test statistics:

- tau3 refers to the t-statistic associated to $H_0: \gamma = 0$ i.e. the presence of a UR. H_1 refers to the absence of said UR.
- phi2 refers to the F-statistic associated with $H_0: \alpha = \beta = \gamma = 0$ ²
- phi3 is also an F-statistic, now associated to $H_0: \beta = \gamma = 0$

Remark that the critical value in both tables can be a little different. This is because they are sensitive to the number of observations in each series. In Table 1, the critical values correspond to those provided directly by R and are associated with $N = 500$, while in Table 2 we give the values for $N = 250$. Since we have 396 data points per series we preferred to refer to the higher critical values but it does not change the analysis done.

Let us examine each series' results, summarized in the following tables:

Table 1: ADF test - 1st regression with drift, deterministic trend and stochastic trend

	gdp	dpi	infl_e	deflator	rate	splong	CV 1pct	CV 5pct	CV 10pct
tau3	-2.150	-2.694	-4.650	-5.197	-1.709	-1.975	-3.980	-3.420	-3.130
phi2	14.496	7.880	7.375	9.097	2.015	3.875	6.150	4.710	4.050
phi3	2.388	3.849	11.061	13.644	2.917	1.997	8.340	6.300	5.360

²As with all F-tests, the alternative hypothesis is that at least one of these coefficients is non-null. Since this is general, we don't explicitly signal the H_1 hereafter.

Table 2: ADF test - 1st regression t statistics

	gdp	dpi	infl_e	deflator	rate	splong
alpha	2.197	2.714	4.004	2.593	0.698	2.101
gamma	-2.150	-2.694	-4.650	-5.197	-1.709	-1.975
beta	2.094	2.577	1.265	1.166	-0.181	1.726
rho	16.326	-11.089	-0.255	14.255	15.719	4.099

Notes: With N=396, critical values at 5%: alpha = 3.09 ; gamma= -3.43 ; beta = 2.79

GDP We have $t_\gamma = -2.15 > -3.42$ we cannot reject the presence of a UR (H_0). We shall note that phi2 ie that all coefficients are null is rejected since $F_{phi2} = 14.496 > 4.71$ while phi3 ($\gamma = \beta = 0$) is not ($F_{phi3} = 2.388 < 6.3$). This suggests the absence of a deterministic trend which is confirmed when assessing the significance of β using its non-standard critical value $|t_\beta| = 2.094 < 2.79$ (nullity, ie H_0 cannot be rejected). These results require us to keep testing the series with the next specification of the test.

Disposable personal income Similarly as before we find $t_\gamma = -2.694 > -3.42$ and we fail to reject H_0 . Regarding the joint nullity tests as before we reject phi2 ($F_{phi2} = 7.88 > 4.71$) and cannot reject phi3 ($F_{phi3} = 3.849 < 6.3$). We also fail to reject the nullity of β as $|t_\beta| = 2.577 < 2.79$ and we shall test this series moving forward.

Inflation expectation We find $t_\gamma = -4.65 < -3.43$ we reject H_0 ie we can't say that the series has a UR. The F-statistics of phi2 and phi3 lead us to reject their null hypothesis: $F_{phi2} = 7.375 > 4.71$, $F_{phi3} = 11.061 > 6.3$, leading us to believe that the series has either a drift and/or a deterministic trend. We, therefore, compare the t-statistics associated with α and β to the standard interest threshold (the critical values below Table 2 are conditional on having a UR). Since $|t_\alpha| = 4.004 > 1.96$ and $|t_\beta| = 1.265 < 1.96$, we fail to reject the presence of a deterministic trend while the drift term is significantly different from zero. We conclude that the series is *stationnary with a constant and without a deterministic trend*.

GDP deflator With $t_\gamma = -5.197 < -3.43$, as before we can reject H_0 indicating that the series is stationary in levels. Since $|t_\alpha| = 2.593 > 1.96$, we reject the nullity of the drift. Finally, since $|t_\beta| = 1.166 < 1.96$ we cannot reject the absence of a deterministic trend. We conclude that the series is *stationary with a constant and without a deterministic trend*

Fed fund rate $t_\gamma = -1.709 > -3.43$, we are in the same situation as the previous series where we cannot reject the existence of a UR. Because we cannot reject phi2 nor phi3 ($F_{phi2} = 2.015 < 4.71$, $F_{phi3} = 2.917 < 6.3$) we need to continue testing this series with the other specifications as we don't reject the existence of a stochastic trend and we cannot reject the nullity of the trend coefficient ($|t_\beta| = 0.181 < 2.79$).

S&P500 Without many surprises for price series, $t_\gamma = -1.975 > -3.43$ and we cannot reject the existence of a UR. Moreover, $F_{phi2} = 3.875 < 4.71$ and $F_{phi3} = 1.997 < 6.3$ (non-rejection of the null for both tests) leads

to conclude that at least one of these coefficients is non-null (note that $|t_\beta| = 1.726 < 2.79$ and thus we cannot reject the nullity of β). We continue testing this series with the second specification of the test.

1.2.2 ADF - Test jointly for stochastic trend and drift

The second specification of the test models $\forall (X_t)_t$:

$$\Delta X_t = \alpha + \gamma X_{t-1} + \sum_{i=1,2,..} \rho_i \Delta X_{t-i} + \varepsilon_t \quad (2)$$

The null hypothesis still refers to $H_0: \gamma = 0$ the presence of a unit root. We use now `type='trend'` in the `ur.df` function to get this estimation. The output of the test is similar to the previous specification and the same remarks on the critical values apply here. Now the test statistics reported refer to:

- tau2 refers to the t-statistic associated to $\gamma = 0$
- phi1 refers to the F-statistic associated to $\alpha = \gamma = 0$

Table 3: ADF test - 2nd regression with drift and stochastic trend

	gdp	dpi	rate	splong	CV 1pct	CV 5pct	CV 10pct
tau2	-0.621	-1.020	-2.412	-1.005	-3.440	-2.870	-2.570
phi1	19.382	8.378	3.014	4.300	6.470	4.610	3.790

Table 4: ADF test - 2nd regression t statistics

	gdp	dpi	rate	splong
alpha	0.983	1.127	1.508	1.258
gamma	-0.621	-1.020	-2.412	-1.005
rho	16.197	-12.106	15.984	3.977

Notes: With N=396, critical values at 5%: alpha = 2.53 ; gamma= -2.88

GDP We fail to reject the main null hypothesis (tau2) as $t_\gamma = -0.621 > -2.88$. Moreover, we do reject the joint nullity of α and γ as $F_{phi1} = 19.382 > 6.470$. Since we cannot reject the nullity of the drift term ($|t_\alpha| = 0.983 < 2.53$), we shall use the last specification of the test on this series.

Disposable personal income We fall in the same situation as with the previous series as tau2 is not rejected $t_\gamma = -1.02 > -2.88$ while phi1 is $F_{phi1} = 8.378 > 6.470$ and α is non-significant ($|t_\alpha| = 1.127 < 2.53$). We therefore also test this series with the last test specification.

Fed fund rate Given that $t_\gamma = -2.412 > -2.88$, we cannot reject the null hypothesis. We then check the F-statistic of the joint test *phi1*: $F_{phi1} = 3.014 < 4.61$: we cannot reject the null suggesting that the series has a UR and no drift. Supporting this, we also find that the drift term is not significantly different from zero as $|t_\alpha| = 1.508 < 2.53$. This leads us to use the third specification of the test.

S&P500 Since $t_\gamma = -1.005 > -2.87$, we cannot reject H_0 . By checking $F_{phi1} = 4.3 < 4.61$ and $|t_\alpha| = 1.258 < 2.53$, we fall in the same case as before where we need to continue testing the series as it seems to have a UR and no drift

1.2.3 ADF - Test for stochastic trend only

The last specification of the test keeps only the stochastic trend, $\forall (X_t)_t$:

$$\Delta X_t = \gamma X_{t-1} + \sum_{i=1,2,..} \rho_i \Delta X_{t-i} + \varepsilon_t \quad (3)$$

The null hypothesis still refers to $H_0: \gamma = 0$ the presence of a unit root and we use `type='none'`. The output of the test is similar to the previous ones but now there is only one test statistic reported referring to the null (τ_1). For this step, we only report the table with the t-statistics as its value for the "gamma" row is identical to the test statistics of τ_1 .

Table 5: ADF test - 3rd regression t statistics

	gdp	dpi	rate	splong
gamma	6.148	3.934	-1.935	2.647
rho	16.306	-12.122	15.950	3.976

Notes: With N=396, critical values at 5%: gamma= -1.95

GDP We have $t_\gamma = 6.148 > -1.95$. We clearly do not reject the presence of a UR (H_0) and conclude that the *series has a unit root with no constant nor time trend*.

Disposable personal income Since $t_\gamma = 3.934 > -1.95$ we do not reject H_0 and conclude that the *series has a unit root with no constant nor time trend*.

Fed fund rate We find $t_\gamma = -1.935 > -1.95$ thus we cannot reject the existence of a UR (this is a close call but seems adequate since we never rejected the UR and R's built-in `order_integration` also indicates a UR). We conclude that the *series has a unit root with no constant nor time trend*.

S&P500 Similarly, we find $t_\gamma = 2.647 > -1.95$ and we cannot reject H_0 and conclude that the *series has a unit root with no constant nor time trend*.

1.2.4 Check stationarity of the series in deltas if UR in levels

Finally, to properly conclude that the previous series with UR are indeed $I(1)$, we need to check that the series of their first differences are stationary (i.e. the series in deltas is $I(0)$). We perform the same ADF procedure to test these transformed series.

Table 6 reports the test results in the first specification of the ADF test. We easily see that all the t_γ (the statistic on τ_3) are sufficiently negative to reject H_0 and conclude that none of the differentiated series has a UR. This

Table 6: ADF test - 1st regression with drift, deterministic trend and stochastic trend for series in deltas

	d_gdp	d_dpi	d_rate	d_splong	CV 1pct	CV 5pct	CV 10pct
tau3	-10.437	-19.070	-7.107	-13.317	-3.980	-3.420	-3.130
phi2	36.317	121.228	16.895	59.110	6.150	4.710	4.050
phi3	54.473	181.842	25.327	88.665	8.340	6.300	5.360

is sufficient to conclude that the series of GDP, of real disposable individual income, of the Fed fund rate, and of the returns of the S&P500 are indeed *integrated of order one*.

1.3 Decomposition of the series

All (monthly³) time series de can be decomposed into the following elements:

$$X_t = \underbrace{\alpha}_{\text{drift}} + \underbrace{\beta \times t}_{\text{deterministic trend}} + \underbrace{\gamma Tt}_{\text{stochastic trend}} + \underbrace{\sum_{i=1}^{11} \rho_i \mathbb{1}_{\text{month}=i}}_{\text{seasonality}} + \underbrace{c_t}_{\text{cyclical component}}$$

Remark that the coefficients in the previous equation need not be the same as in the ADF test (the variable on the left-hand side is now in levels while AD tests the series in deltas).

We will apply this decomposition to stationary series. Therefore, the stochastic term $\gamma Tt = 0$ for all series.

Using R's built-in function `stl`⁴, we can decompose the time series. The results are plotted in Figure 1.

1.3.1 Estimation of the parameters in the stationary series

We perform we estimate the following regression to test the significance of the deterministic trend, drift and seasonal components:

$$X_t = \alpha + \beta \times t + \sum_{i=1}^{11} \rho_i \mathbb{1}_{\text{month}=i} + c_t$$

The results of this estimation are found in Table 7 for the series that are $I(0)$ in levels, and in Table 8 for the series of first differences of the series that were not stationary

The coefficients on trend correspond to $\hat{\beta}$, those on the variables starting with M are $\hat{\rho}_i$. Since there is a monthly dummy less than the number of months, the coefficient on `Contsnat` encompasses both α and the coefficient on the last month. We tried to change the dummy that is ommitted and it does not change the results in terms of significance.

³One should adapt the number of indicators in the seasonal components according to the data frequency. To avoid multicollinearity issues, the number of dummies must be one less than the frequency of the series: here we have only 11 indicators.

⁴Seasonal Decomposition of Time Series by Loess

Table 7: OLS decomposition of I(0) series

	<i>Dependent variable:</i>	
	infl_e (1)	deflator (2)
trend	0.001** (0.0003)	0.002** (0.001)
M1	0.051 (0.166)	0.083 (0.347)
M2	0.050 (0.166)	0.097 (0.347)
M3	0.192 (0.166)	0.112 (0.347)
M4	0.239 (0.166)	0.127 (0.347)
M5	0.311* (0.166)	0.109 (0.347)
M6	0.262 (0.166)	0.091 (0.347)
M7	0.180 (0.166)	0.073 (0.347)
M8	0.252 (0.166)	0.016 (0.347)
M9	0.196 (0.166)	−0.041 (0.347)
M10	0.174 (0.166)	−0.098 (0.347)
M11	0.055 (0.166)	−0.049 (0.347)
Constant	2.745*** (0.132)	1.899*** (0.276)
Observations	396	396
R ²	0.036	0.018
Adjusted R ²	0.006	−0.012
Residual Std. Error (df = 383)	0.676	1.409
F Statistic (df = 12; 383)	1.208	0.600

Note:

*p<0.1; **p<0.05; ***p<0.01

Table 8: OLS decomposition of the first difference of I(1) series

	<i>Dependent variable:</i>			
	d_gdp (1)	d_dpi (2)	d_rate (3)	d_splong (4)
trend	−0.00000 (0.00000)	−0.00000 (0.00001)	0.0003*** (0.0001)	−0.00000 (0.00002)
M1	0.0001 (0.001)	−0.005 (0.005)	0.019 (0.046)	−0.004 (0.009)
M2	0.0004 (0.001)	−0.007 (0.005)	0.013 (0.045)	−0.008 (0.009)
M3	0.0004 (0.001)	0.001 (0.005)	0.025 (0.045)	−0.013 (0.009)
M4	0.0004 (0.001)	−0.006 (0.005)	0.014 (0.045)	0.001 (0.009)
M5	0.002 (0.001)	−0.005 (0.005)	0.055 (0.045)	−0.004 (0.009)
M6	0.002 (0.001)	−0.007 (0.005)	0.083* (0.045)	−0.009 (0.009)
M7	0.002 (0.001)	−0.005 (0.005)	0.051 (0.045)	−0.007 (0.009)
M8	0.001 (0.001)	−0.006 (0.005)	0.048 (0.045)	−0.013 (0.009)
M9	0.001 (0.001)	−0.006 (0.005)	0.036 (0.045)	−0.017* (0.009)
M10	0.001 (0.001)	−0.006 (0.005)	−0.021 (0.045)	−0.017* (0.009)
M11	0.00001 (0.001)	−0.005 (0.005)	0.011 (0.045)	0.007 (0.009)
Constant	0.003*** (0.001)	0.007** (0.004)	−0.091** (0.036)	0.014** (0.007)
Observations	395	395	395	395
R ²	0.021	0.019	0.048	0.037
Adjusted R ²	−0.010	−0.012	0.018	0.007
Residual Std. Error (df = 382)	0.004	0.018	0.185	0.036
F Statistic (df = 12; 382)	0.668	0.602	1.598*	1.230

Note:

*p<0.1; **p<0.05; ***p<0.01

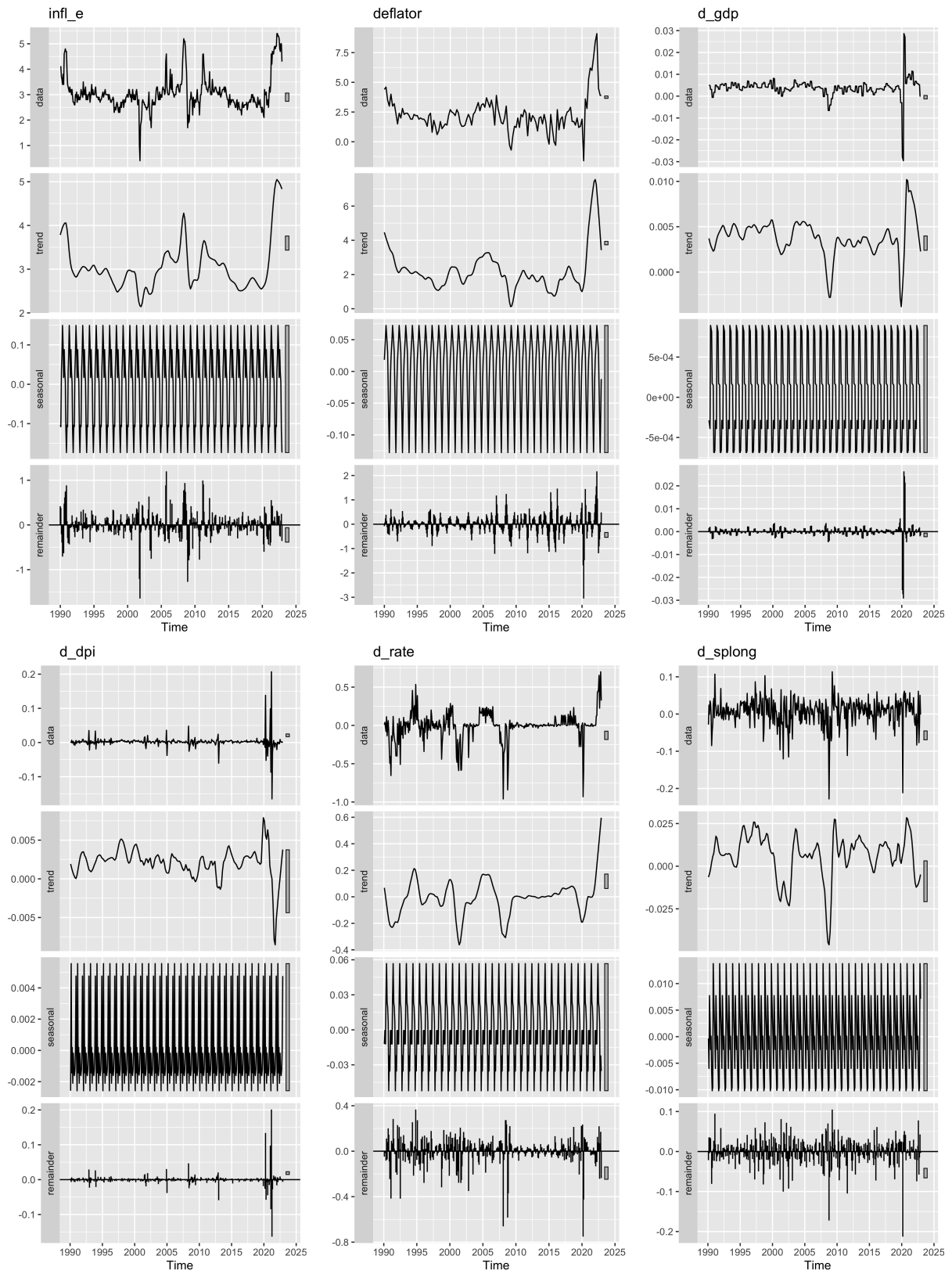


Figure 1: Time series decomposition of stationary series (levels and deltas)

1.4 Cyclical component

To assess the behavior of the cyclical component, we detrend the series, subtract the coefficient of the constant and remove the seasonal component when the coefficients are significant. After extracting these components, we would like to model them with appropriate $ARMA(p, q)$ models.

To do so, we first produce PACF (Figure 2) and ACF graphs (Figure 3) to know respectively the maximum values that p and q could take, \bar{p} , \bar{q} . The blue dotted lines on the figures are the 95% confidence intervals. \bar{p} , \bar{q} are given by the last period where the PACF or the ACF are significantly different from zero. We were surprised by the rather long memory of some of the processes, notably of the inflation expectation, the GDP deflator, and the first difference of the Fed fund rate series but we double checked and the series are stationary.

We then find the best parameters for each model by minimizing the BIC (an informational criteria) for each combination of $p \leq \bar{p}$ and $q \leq \bar{q}$. This will allow us to find a combination of both types of models (AR and MA) such that we capture well the past with less parameters to estimate. To do so, we use R's `critMatrix` function with the option `criteria = 'bic'`. Importantly, this operation is slow in particular when \bar{p} , \bar{q} are both large. In the case of the inflation expectation and the first difference of the Fed fund rate, their values were so large that R does not give any answer and reports an error. To solve this, we decided to take smaller values of \bar{p} . Indeed, by looking at the PACF graphs (Figure 2), we see that in both cases there is a single lag down the line that is significantly different from zero after several ones falling in the CI (lag 13 for *infl-e* and lag 15 for *d-rate*). We took the previous significant lag as \bar{p} to be able to get the information criteria with the rationale that those "extreme" significant lags can be due to just luck, and that the combination of AR and MA models should allow to reduce the number of parameters.

We present the outputs of these comparisons tables per series in Appendix 5.2 because they are a lot and they were making hard to read this section. Remark that the placement of the ps and qs can vary from one table to the next due to formatting issues.

2 Canonical VAR model application

Table 9: Canonical VAR in levels - Identify order

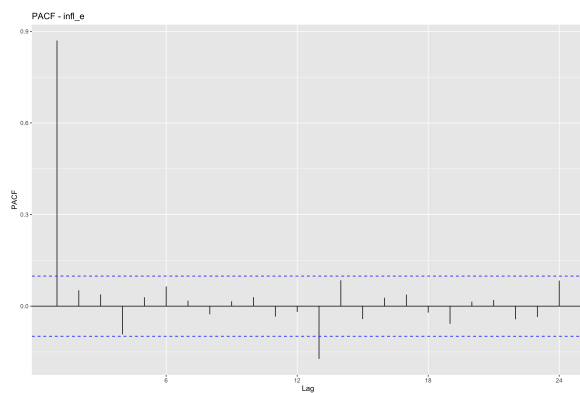
	1	2	3	4	5	6	7	8	9	10
AIC(n)	-10.01	-10.56	-10.59	-10.73	-10.88	-10.87	-10.93	-11.01	-11.00	-10.98
HQ(n)	-9.96	-10.48	-10.47	-10.57	-10.68	-10.64	-10.66	-10.71	-10.66	-10.60
SC(n)	-9.89	-10.35	-10.29	-10.33	-10.38	-10.29	-10.25	-10.25	-10.14	-10.03
FPE(n)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 10: Level VAR - Estimation

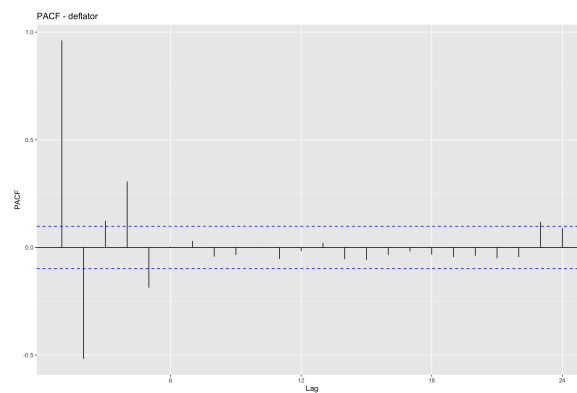
	<i>Dependent variable:</i>		
	deflator	unempl	splong
deflator.l1	1.830*** (0.051)	-0.199 (0.124)	0.026*** (0.008)
unempl.l1	0.120*** (0.022)	0.956*** (0.053)	0.003 (0.004)
splong.l1	0.556* (0.321)	-4.968*** (0.786)	1.174*** (0.053)
deflator.l2	-0.819*** (0.101)	-0.317 (0.246)	-0.042** (0.017)
unempl.l2	-0.117*** (0.031)	-0.131* (0.075)	-0.004 (0.005)
splong.l2	-0.553 (0.506)	5.816*** (1.237)	-0.246*** (0.083)
deflator.l3	-0.894*** (0.108)	1.008*** (0.264)	0.010 (0.018)
unempl.l3	0.014 (0.031)	0.182** (0.076)	0.002 (0.005)
splong.l3	0.149 (0.521)	-0.992 (1.275)	0.137 (0.085)
deflator.l4	1.526*** (0.113)	-0.716** (0.277)	0.019 (0.019)
unempl.l4	-0.013 (0.031)	-0.150** (0.076)	0.003 (0.005)
splong.l4	0.428 (0.523)	-0.516 (1.278)	-0.012 (0.086)
deflator.l5	-0.653*** (0.111)	-0.130 (0.271)	-0.021 (0.018)
unempl.l5	0.001 (0.031)	0.092 (0.076)	-0.008* (0.005)
splong.l5	-0.027 (0.527)	1.417 (1.288)	0.063 (0.086)
deflator.l6	-0.498*** (0.107)	0.609** (0.262)	-0.003 (0.018)
unempl.l6	0.004 (0.031)	-0.105 (0.075)	0.006 (0.005)
splong.l6	-1.048** (0.527)	-1.069 (1.290)	-0.302*** (0.086)
deflator.l7	0.834*** (0.103)	-0.476* (0.252)	0.019 (0.017)
unempl.l7	0.081*** (0.030)	0.088 (0.073)	-0.0004 (0.005)
splong.l7	0.954* (0.526)	-0.455 (1.287)	0.261*** (0.086)
deflator.l8	-0.350*** (0.054)	0.153 (0.131)	-0.010 (0.009)
unempl.l8	-0.075*** (0.021)	0.039 (0.052)	-0.001 (0.004)
splong.l8	-0.428 (0.340)	0.746 (0.832)	-0.078 (0.056)
const	-0.268* (0.149)	0.490 (0.365)	0.015 (0.024)
Adjusted R ²	0.977	0.913	0.997
Residual Std. Error (df = 363)	0.210	0.515	0.034
F Statistic (df = 24; 363)	684.697***	170.695***	5,592.560***

Note:

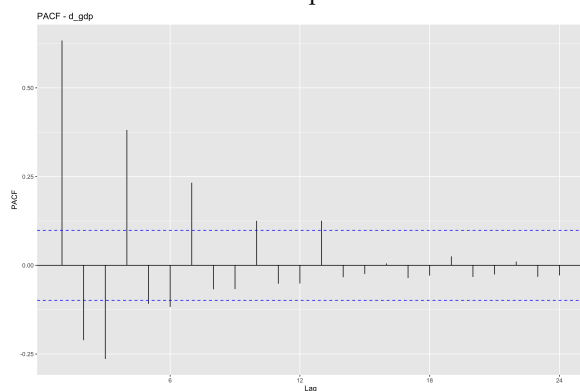
*p<0.1; **p<0.05; ***p<0.01



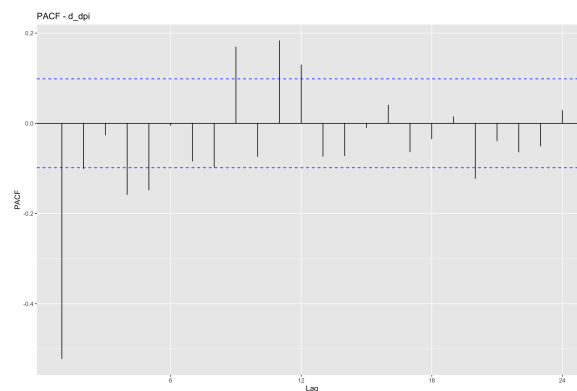
Inflation expectation



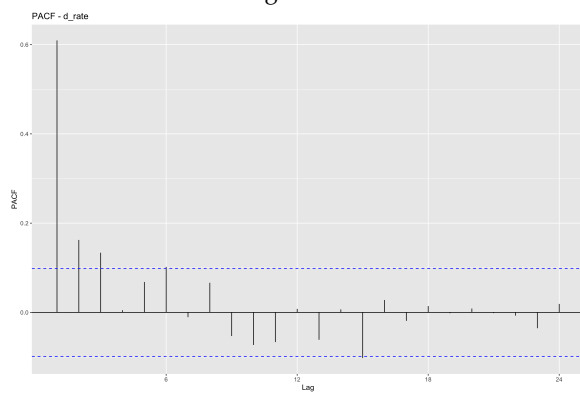
GDP deflator



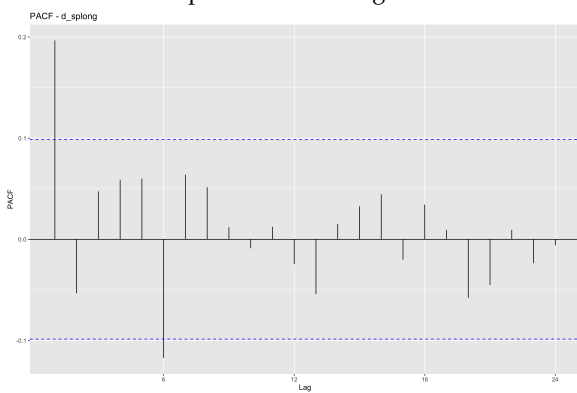
GDP growth rate



Real disposable income growth rate



Fed fund rate



Returns of S&P500

Figure 2: PACFs

Notes: All the series plotted are stationary: inflation expectation and GDP deflator series are in levels while the rest are first differences.

3 Cointegration theory

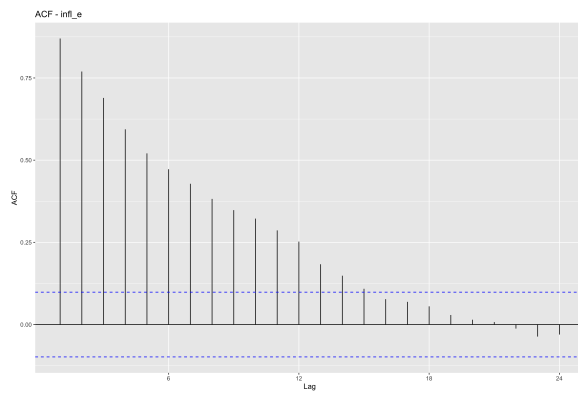
4 Impulse Response Analysis

4.1 Canonical IRF

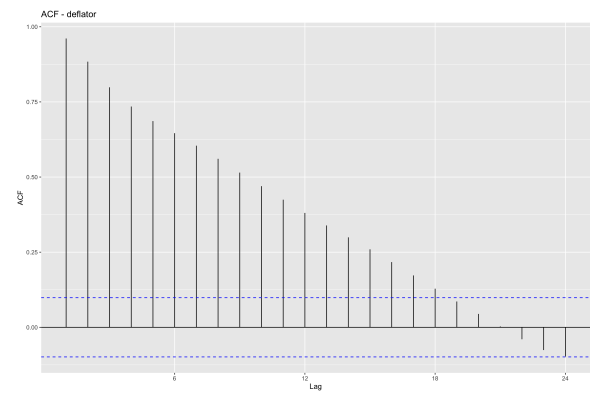
4.2 Structural IRF

5 Introduce non-linearities

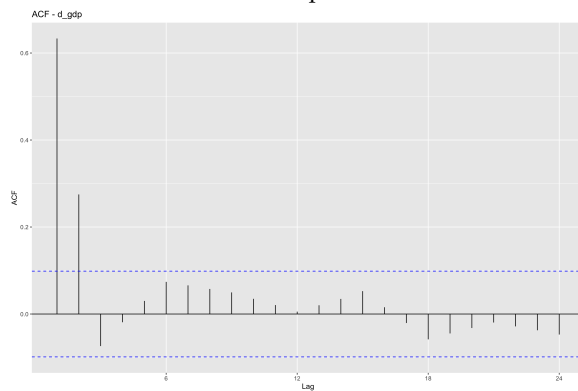
5.1 Markov-switching model



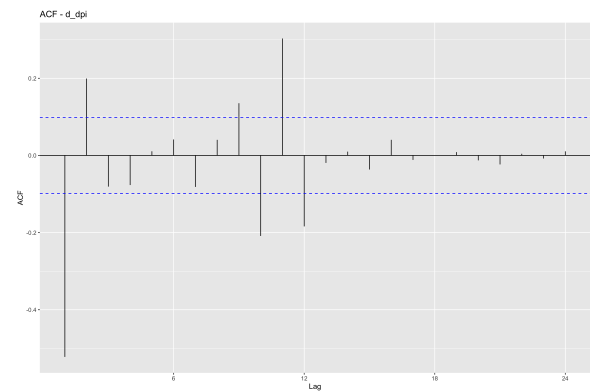
Inflation expectation



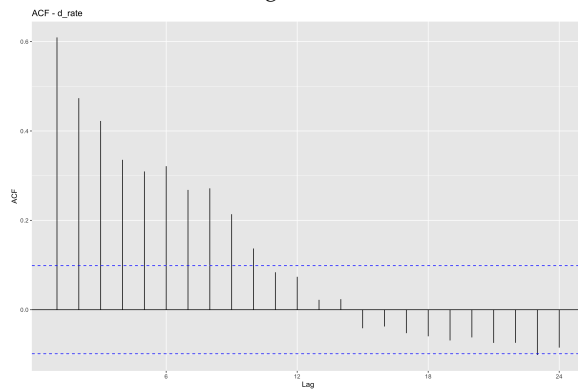
GDP deflator



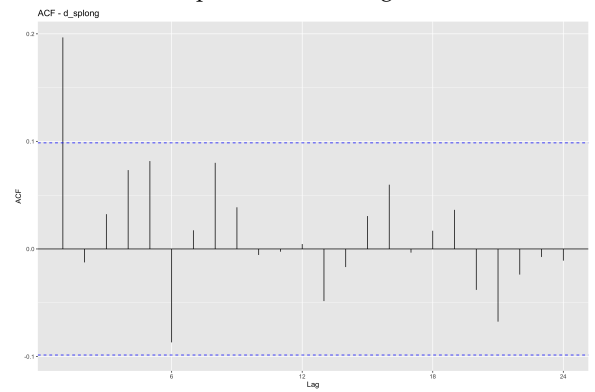
GDP growth rate



Real disposable income growth rate



Fed fund rate



Returns of S&P500

Figure 3: ACFs

Notes: All the series plotted are stationary: inflation expectation and GDP deflator series are in levels while the rest are first differences.

Section 1 - Additional tables

Appendix A Code - Data Cleaning

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
```

Table 11: Information criteria on the parameters of ARMA for infl-e

	p=0	p=1	p=2	p=3	p=4
q=0	808.438	236.035	239.455	244.671	246.751
q=1	517.737	239.263	239.841	244.875	252.676
q=2	413.415	245.105	244.731	249.439	254.721
q=3	348.231	247.220	249.095	254.601	254.129
q=4	298.207	252.154	246.078	259.082	265.384
q=5	292.940	254.133	246.824	251.856	258.027
q=6	289.687	259.915	252.000	257.838	262.286
q=7	278.591	265.863	257.687	263.809	264.647
q=8	275.739	270.725	262.675	266.930	270.434
q=9	281.720	276.114	264.761	270.662	274.023
q=10	284.970	277.918	270.633	271.776	278.872
q=11	289.458	283.842	275.724	276.490	283.533
q=12	284.137	277.990	279.720	282.047	289.110
q=13	286.646	283.125	279.067	285.024	290.893
q=14	288.837	288.879	285.013	290.070	288.811
q=15	292.928	294.809	290.843	306.880	298.582

Table 12: Information criteria on the parameters of ARMA for infl-e

	p=0	p=1	p=2	p=3	p=4
q=0	808.438	236.035	239.455	244.671	246.751
q=1	517.737	239.263	239.841	244.875	252.676
q=2	413.415	245.105	244.731	249.439	254.721
q=3	348.231	247.220	249.095	254.601	254.129
q=4	298.207	252.154	246.078	259.082	265.384
q=5	292.940	254.133	246.824	251.856	258.027
q=6	289.687	259.915	252.000	257.838	262.286
q=7	278.591	265.863	257.687	263.809	264.647
q=8	275.739	270.725	262.675	266.930	270.434
q=9	281.720	276.114	264.761	270.662	274.023
q=10	284.970	277.918	270.633	271.776	278.872
q=11	289.458	283.842	275.724	276.490	283.533
q=12	284.137	277.990	279.720	282.047	289.110
q=13	286.646	283.125	279.067	285.024	290.893
q=14	288.837	288.879	285.013	290.070	288.811
q=15	292.928	294.809	290.843	306.880	298.582

Table 13: Information criteria on the parameters of ARMA for d-gdp

	q=0	q=1	q=2
p=0	-3,196.342	-3,335.595	-3,556.284
p=1	-3,393.003	-3,395.734	-3,550.426
p=2	-3,404.863	-3,405.710	-3,544.540
p=3	-3,427.848	-3,456.107	-3,540.254
p=4	-3,484.507	-3,481.070	-3,534.314
p=5	-3,483.226	-3,478.455	-3,528.379
p=6	-3,483.225	-3,488.833	-3,523.609
p=7	-3,500.495	-3,495.556	-3,517.633
p=8	-3,496.418	-3,490.948	-3,511.656
p=9	-3,492.678	-3,490.332	-3,507.291
p=10	-3,493.990	-3,488.534	-3,501.325
p=11	-3,489.174	-3,483.437	-3,495.360
p=12	-3,484.524	-3,482.220	-3,489.569
p=13	-3,485.664	-3,480.011	-3,483.611

Table 14: Information criteria on the parameters of ARMA for d-dpi

	q=0	q=1	q=2	q=3	q=4	q=5	q=6	q=7	
p=0	-2,040.006	-2,159.574	-2,155.516	-2,167.273	-2,161.611	-2,157.417	-2,151.693	-2,150.189	-2
p=1	-2,159.554	-2,158.813	-2,154.793	-2,161.479	-2,159.258	-2,155.433	-2,145.472	-2,149.050	-2
p=2	-2,157.605	-2,161.222	-2,155.416	-2,156.642	-2,163.780	-2,162.047	-2,143.475	-2,151.355	-2
p=3	-2,151.898	-2,155.614	-2,152.945	-2,164.239	-2,159.070	-2,154.684	-2,151.312	-2,146.238	-2
p=4	-2,155.878	-2,157.819	-2,158.010	-2,160.097	-2,159.219	-2,165.859	-2,150.022	-2,155.182	-2
p=5	-2,158.597	-2,153.039	-2,147.634	-2,154.626	-2,166.740	-2,161.024	-2,155.178	-2,151.351	-2
p=6	-2,152.629	-2,147.327	-2,141.825	-2,149.321	-2,155.802	-2,154.798	-2,151.579	-2,146.862	-2
p=7	-2,149.433	-2,144.239	-2,152.828	-2,143.108	-2,148.209	-2,142.850	-2,155.557	-2,150.118	-2
p=8	-2,147.169	-2,152.831	-2,143.483	-2,143.202	-2,149.709	-2,144.433	-2,149.815	-2,135.057	-2
p=9	-2,152.503	-2,145.737	-2,147.194	-2,141.698	-2,148.990	-2,138.511	-2,132.540	-2,140.389	-2
p=10	-2,148.638	-2,141.516	-2,141.326	-2,139.324	-2,147.074	-2,144.611	-2,144.907	-2,142.275	-2
p=11	-2,155.935	-2,153.152	-2,154.028	-2,149.088	-2,143.284	-2,137.544	-2,132.018	-2,126.041	-2
p=12	-2,156.684	-2,151.663	-2,149.126	-2,143.180	-2,146.342	-2,141.996	-2,126.053	-2,123.091	-2

Table 15: Information criteria on the parameters of ARMA for d-rate

	p=0	p=1	p=2	p=3	p=4	p=5	p=6
q=0	-220.437	-398.928	-405.474	-407.769	-401.798	-398.623	-397.967
q=1	-336.159	-412.584	-409.586	-403.731	-401.328	-395.918	-396.298
q=2	-357.789	-409.855	-404.200	-398.327	-395.988	-397.849	-392.506
q=3	-381.064	-403.959	-398.280	-398.236	-391.210	-385.390	-388.208
q=4	-385.108	-399.279	-393.396	-391.439	-385.366	-380.777	-386.024
q=5	-380.280	-393.931	-394.569	-386.948	-379.522	-379.390	-384.388
q=6	-381.165	-394.107	-394.053	-394.640	-389.631	-382.456	-378.467
q=7	-375.560	-388.690	-387.899	-389.048	-375.038	-371.914	-368.204
q=8	-376.117	-391.305	-386.325	-380.439	-374.871	-372.999	-367.089
q=9	-379.145	-386.439	-380.464	-374.571	-369.119	-368.622	-367.891
q=10	-381.673	-380.470	-380.584	-375.879	-369.271	-367.834	-357.236

Table 16: Information criteria on the parameters of ARMA for d-splong

	q=0	q=1
p=0	-1,498.418	-1,509.363
p=1	-1,508.029	-1,503.624
p=2	-1,503.196	-1,498.958
p=3	-1,498.133	-1,493.386
p=4	-1,493.534	-1,487.872
p=5	-1,488.927	-1,486.959
p=6	-1,488.534	-1,483.508

```

4 Financial Econometrics - Empirical Applications d
5 Data Gathering and Data Cleaning
6
7 @author: nataliacardenasf
8 """
9
10 import pandas as pd
11 import os
12 import datetime
13
14 from fredapi import Fred
15
16 os.chdir('/Users/nataliacardenasf/Documents/GitHub/PROJECTS_AP_FE/FinancialEconometrics1')
17
18 ### Initialize FRED API
19 fred = Fred(api_key='23edc2b1b61e17c07b83a97e7abfc02b')
20
21 ### Import all the data
22
23 # S&P 500
24 sp500 = pd.DataFrame(fred.get_series('SP500')) #daily close, NSA, Index
25 sp500.columns= ['sp500']
26
27 #Inflation expectations from survey UMich
28 infl_e = pd.DataFrame(fred.get_series('MICH')) #monthly, NSA, median expected in % over next
29         12 mo
30 infl_e.columns= ['infl_e']
31
32 #ICE BofA US Corporate Index Total Return Index
33 corp_debt = pd.DataFrame(fred.get_series('BAMLCOAOCMTRIV')) #daily, close, NSA, Index
34 corp_debt.columns= ['corp_debt']
35
36
37 #MP rate
38 rate = pd.DataFrame(fred.get_series('DFF')) #daily, 7-Day, NSA, %
39 rate.columns = ['rate']

```

```

40
41 #Deflator
42 deflator = pd.DataFrame(fred.get_series('A191RI1Q225SBEA')) #Q, SA Annual Rate
43 deflator.columns = ['deflator']
44
45 #Unemployment
46 unempl = pd.DataFrame(fred.get_series('UNRATENSA')) #monthly, NSA, %
47 unempl.columns=['unempl']
48
49 # GDP
50 gdp = pd.DataFrame(fred.get_series('GDP')) # quarterly, Billions of Dollars, SA Annual Rate
51 gdp.columns = ['gdp']
52
53 # RPI (Real Personal Income)
54 rpi = pd.DataFrame(fred.get_series('RPI')) # monthly, SA rate, deflated
55 rpi.columns = ['rpi']
56
57 # Real Personal Disposable Income
58 dpi = pd.DataFrame(fred.get_series('DSPIC96')) # monthly, SA annaul rate, chained 2017 USD
59 dpi.columns = ['dpi']
60
61 # Manufacturing Sector
62 manufacturing = pd.DataFrame(fred.get_series('MPU9900063')) # annual, NSA => avoid this
63 manufacturing.columns = ['manufacturing']
64
65 fred.search("MPU9900063").T #this function gives of the info on every series
66
67
68 ### Resample into monthly data
69 sp500 = sp500.resample('1M').mean(numeric_only=True)
70 infl_e = infl_e.resample('1M').mean(numeric_only=True)
71 corp_debt = corp_debt.resample('1M').mean(numeric_only=True)
72 rate = rate.resample('1M').mean(numeric_only=True)
73 deflator = deflator.resample('1M').mean(numeric_only=True)
74 unempl = unempl.resample('1M').mean(numeric_only=True)
75 gdp = gdp.resample('1M').mean(numeric_only=True)
76 rpi = rpi.resample('1M').mean(numeric_only=True)
77 dpi = dpi.resample('1M').mean(numeric_only=True)
78 manufacturing = manufacturing.resample('1M').mean(numeric_only=True)
79
80 dta = [infl_e, rate, sp500, corp_debt, deflator, unempl, gdp, rpi, dpi, manufacturing]
81
82 ### Slice the df to relevant period
83 #Find common time span
84 min_date = max([min(i.index) for i in dta])
85 max_date = min([max(i.index) for i in dta])
86 print(min_date, max_date)
87

```

```

88 #Let us work on monthly data for the 1990-2022 period
89 start = datetime.datetime(1990,1,1)
90 end= datetime.datetime(2022,12,31)
91
92 ## SP500 series is too short, I am taking it from Yahoo Finance
93 import yfinance as yf
94 splong = yf.download('^GSPC', start=start,end=end)['Adj Close'].resample('M').mean(
    numeric_only=True)
95 splong = pd.DataFrame(splong)
96 type(splong)
97 splong.rename(columns={"Adj Close":'splong'}, inplace=True)
98
99 ##Get a single DF
100 dta.append(splong)
101 for i in range(len(dta)): #we had some indexes at end of month, others at 1st of month:
    harmonize to 1st each month
102     df = dta[i]
103     df.index = [pd.datetime(x.year, x.month, 1) for x in df.index.tolist()]
104     dta[i] = df.loc[start:end,:]
105 dta# we're good now
106 #merge into 1 df, 1 series per column
107 monthly = pd.concat(dta, axis=1)
108 #interpolate missing months for deflator data (Q): uses midpoints ie assumes that each month
    in the quarter contributes in the same fashion to the increase QoQ
109 m1 = monthly.interpolate(method = 'linear', limit_direction = 'forward')
110
111
112 m1.to_csv("DATA/data.csv")

```

Appendix B Code - Analysis 1

```

1 ##%% FE 1 v3
2 ##%% @ncardenasfrias
3
4 # Load necessary packages and set personal path to documents
5 pacman::p_load(data.table, tseries, smoots, dplyr, bootUR, urca, gridExtra, tidyverse, gplots,
    xts, stargazer, forecast, ggplot2, vars)
6
7 setwd('/Users/nataliacardenasf/Documents/GitHub/PROJECTS_AP_FE/FinancialEconometrics1')
8
9 #####
10 #O. Preprocessing
11 #####
12
13 #Load the dataset, data gathering and cleaning done in Python
14 df = fread("DATA/data.csv")
15 as.data.table(df)
16 #change few names of columns to got something more clear

```

```

17 names(df)[1] = 'Date'
18
19
20 #Apply log to variables for which it makes sense ie prices and quantities
21 df$sp500 = log(df$sp500) #get the stock market return
22 df$splong = log(df$splong)
23 df$corp_debt = log(df$corp_debt) #return corporate debt (see index definition)
24 df$gdp = log(df$gdp)
25 df$rpi = log(df$rpi)
26 df$dpi = log(df$dpi)
27
28
29 #convert into TS
30 allts = list()
31 numeric_cols = df[, sapply(df, is.numeric)&names(df)!= 'Date'] #identifies the right columns
32 for(col in names(numeric_cols)){
33   allts[[col]] = ts(df[[col]], start= c(year(df$Date[1]), month(df$Date[1])), frequency=12)
34 }
35
36 allts[['sp500']] = NULL #remove short SP500, use YahooFinance series that has the 90's and 00'
   s data
37 allts[['Date']] = NULL # R was making regressions on the date column :/
38 #remove columns we ended up not using in the final version
39 allts[['manufacturing']] = NULL # rather not have data that was initially yearly
40 allts[['rpi']] = NULL # redundant with dpi
41 allts[['unempl']] = NULL # did not lead anywhere
42 allts[['corp_debt']] = NULL # we end up not using it and the UR test are weird
43
44
45 #Reorganise the list
46 desired_order = c('gdp', 'dpi', 'infl_e', 'deflator', 'rate', 'splong')
47 allts = allts[desired_order]
48 names = c('GDP', "Disposable Income", 'Inflation expectation', 'PIB deflator', 'Fed rate', '
   SP500')
49 rm(df, desired_order)
50
51 #Remove outliers => IT DOES NOT CHANGE ANYTHING
52 # tsclean(allts$infl_e, iterate = 2, lambda =NULL)
53 # tsclean(allts$deflator, iterate = 2, lambda =NULL)
54 # tsclean(allts$unempl, iterate = 2, lambda =NULL)
55 # tsclean(allts$rate, iterate = 2, lambda =NULL)
56 # tsclean(allts$splong, iterate = 2, lambda =NULL)
57 # tsclean(allts$corp_debt, iterate = 2, lambda =NULL)
58
59
60
61 #####
62 #1.UR TEST - ADF, full procedure

```

```

63 #####
64
65 ### ADF 1st regression: deterministic trend + drift
66 results_adf_trend = list()
67 for (var_name in names(allts)) {
68     result = ur.df(allts[[var_name]], type = "trend", selectlags = "BIC")
69     results_adf_trend[[var_name]] = summary(result)
70 }
71 #export summary adf
72 trend_test = cbind(t(results_adf_trend$gdp@teststat), t(results_adf_trend$dpi@teststat),
73                    t(results_adf_trend$infl_e@teststat), t(results_adf_trend$deflator@teststat),
74                    t(results_adf_trend$rate@teststat), t(results_adf_trend$splong@teststat),
75                    results_adf_trend$infl_e@cval)
76 colnames(trend_test) = c(names(results_adf_trend), "CV 1pct", "CV 5pct", "CV 10pct")
77 stargazer(trend_test, type='text')
78 # stargazer(trend_test, out='TABLES/adf_trend.tex', label= 'tab:adftrend_hyp', title = "ADF
79     test - 1st regression with drift, deterministic trend and stochastic trend")
80
81 #export all t-values on coefficients, harder because output format
82 #create empty df with columns = rhs variable in ADF
83 t_values_table = data.frame(matrix(nrow = 0, ncol = length(c('Intercept', 'z.lag.1', 'tt', 'z.
84     diff.lag'))))
85 colnames(t_values_table) = c('alpha', 'gamma', 'beta', 'rho')
86 #iterate over the summaries and extract the t stats for each series
87 for (i in names(results_adf_trend)){
88     j = results_adf_trend[[i]]@testreg$coefficients[, 't value']
89     tab = as.data.frame(j)
90     row = c(tab$j[1], tab$j[2], tab$j[3], tab$j[4])
91     t_values_table[nrow(t_values_table) + 1,] = row
92 }
93 row.names(t_values_table) = names(results_adf_trend) # call the rows as the series
94 t_values_table = t(t_values_table) #transpose the table, looks better
95 stargazer(t_values_table, type='text') #all ok
96 # stargazer(t_values_table, out="TABLES/adf_tstats_trend.tex", title="ADF test - 1st regression
97     t statistics",
98     notes = '\\footnotesize Notes: With N=396, critical values at 5\\%: alpha = 3.09 ;
99     gamma= -3.43 ; beta = 2.79',
100     label='tab:tstat_trend')
101
102
103
104
105 ### ADF 2nd regression: drift
106 results_adf_drift = list()

```

```

107 for (var_name in names(adf_v2)) {
108   result = ur.df(adf_v2[[var_name]], type = "drif", selectlags = "BIC")
109   results_adf_drift[[var_name]] = summary(result)
110 }
111 #export summary adf
112 drift_test = cbind(t(results_adf_drift$gdp@teststat), t(results_adf_drift$dpi@teststat),
113                   t(results_adf_drift$rate@teststat), t(results_adf_drift$spplong@teststat),
114                   results_adf_drift$spplong@cval)
115 colnames(drift_test) = c(names(results_adf_drift), "CV 1pct", "CV 5pct", "CV 10pct")
116 stargazer(drift_test, type='text')
117 # stargazer(drift_test, out='TABLES/adf_drift.tex', label= 'tab:adfdrift_hyp',title = "ADF
    test - 2nd regression with drift and stochastic trend")
118
119 #export all t-values on coefficients, harder because output format
120 #create empty df with columns = rhs variable in ADF
121 t_values_table2 = data.frame(matrix(nrow = 0, ncol = length(c('Intercept', 'z.lag.1', 'z.diff.
    lag'))))
122 colnames(t_values_table2) = c('alpha', 'gamma', 'rho')
123 #iterate over the summaries and extract the t stats for each series
124 for (i in names(results_adf_drift)){
125   j = results_adf_drift[[i]]@testreg$coefficients[, 't value']
126   tab = as.data.frame(j)
127   row = c(tab$j[1], tab$j[2], tab$j[3])
128   t_values_table2[nrow(t_values_table2) + 1,] = row
129 }
130 row.names(t_values_table2) = names(results_adf_drift) # call the rows as the series
131 t_values_table2 = t(t_values_table2) #transpose the table, looks better
132 stargazer(t_values_table2,type='text') #all ok
133 # stargazer(t_values_table2,out="TABLES/adf_tstats_drift.tex", title="ADF test - 2nd
    regression t statistics",
134 #           notes = '\\footnotesize Notes: With N=396, critical values at 5\\%: alpha = 2.53 ;
    gamma= -2.88',
135 #           label='tab:tstat_drift')
136
137
138
139 adf_v3 = adf_v2 #series to keep testing (They are the same but oh well, I had a different
    pipeline with the other series)
140 names(adf_v3) = names(adf_v2)
141
142
143 ### ADF 3rd regression: UR only
144 results_adf_none = list()
145 for (var_name in names(adf_v3)) {
146   result = ur.df(adf_v3[[var_name]], type = "none", selectlags = "BIC")
147   results_adf_none[[var_name]] = summary(result)
148 }
149 #export summary adf

```

```

150 none_test = cbind(t(results_adf_none$gdp@teststat),
151                   t(results_adf_none$dpi@teststat), t(results_adf_none$rate@teststat),
152                   t(results_adf_none$splong@teststat), results_adf_none$rate@cval)
153 colnames(none_test) = c(names(results_adf_none), "CV 1pct", "CV 5pct", "CV 10pct")
154 stargazer(none_test, type='text')
155 # stargazer(none_test, out='TABLES/adf_none.tex', label= 'tab:adfnone_hyp',title = "ADF test -
    3rd regression with stochastic trend")
156
157
158 #export all t-values on coefficients, harder because output format
159 #create empty df with columns = rhs variable in ADF
160 t_values_table3 = data.frame(matrix(nrow = 0, ncol = length(c('z.lag.1', 'z.diff.lag'))))
161 colnames(t_values_table3) = c('gamma', 'rho')
162 #iterate over the summaries and extract the t stats for each series
163 for (i in names(results_adf_none)){
164     j = results_adf_none[[i]]@testreg$coefficients[, 't value']
165     tab = as.data.frame(j)
166     row = c(tab$j[1], tab$j[2], tab$j[3])
167     t_values_table3[nrow(t_values_table3) + 1,] = row
168 }
169 row.names(t_values_table3) = names(results_adf_none) # call the rows as the series
170 t_values_table3 = t(t_values_table3) #transpose the table, looks better
171 stargazer(t_values_table3,type='text') #all ok
172 # stargazer(t_values_table3,out="TABLES/adf_tstats_none.tex", title="ADF test - 3rd regression
    t statistics",
173 #         notes = '\\footnotesize Notes: With N=396, critical values at 5\\%: gamma= -1.95',
174 #         label='tab:tstat_none')
175
176
177
178 ## Get deltas
179 deltas = list (diff(allts$gdp), diff(allts$dpi),
180               diff(allts$rate),diff(allts$splong))
181 names(deltas) = list("d_gdp","d_dpi","d_rate", "d_splong")
182
183 ### CHECK THAT DELTAS ARE I(0), I am using same code as in levels
184 results_adf_trend = list()
185 for (var_name in names(deltas)) {
186     result = ur.df(deltas[[var_name]], type = "trend", selectlags = "BIC")
187     results_adf_trend[[var_name]] = summary(result)
188 }
189 #export summary adf
190 trend_deltas = cbind(t(results_adf_trend$d_gdp@teststat), #t(results_adf_trend$d_rpi@teststat),
191                      t(results_adf_trend$d_dpi@teststat), t(results_adf_trend$d_rate@teststat),
192                      t(results_adf_trend$d_splong@teststat), results_adf_trend$d_rate@cval)
193 colnames(trend_deltas) = c(names(results_adf_trend), "CV 1pct", "CV 5pct", "CV 10pct")
194 stargazer(trend_deltas, type='text')
195 #stargazer(trend_deltas, out='TABLES/adf_deltas.tex', label= 'tab:adfdeltas_hyp',title = "ADF

```



```

    test - 1st regression with drift, deterministic trend and stochastic trend for series in
    deltas")

196
197
198
199
200
201 #####
202 #2. Decompositions in levels
203 #####
204 # use stl function to perform the decompositions and then OLS to estimate parameters
205
206 ### Series in levels that are stationary.
207 i0_levels = list(allts$infl_e, allts$deflator) #from adf!
208 names(i0_levels) = c('infl_e', "deflator")
209
210 decompositions = list() #store the decompositions
211 dec_graphs= list() #store the decomposition graphs
212 deseasonalized_ts = list() #store the deseasonalized ts, if seasonality ends up being
    important
213 for (ts in 1:length(i0_levels)) {
214     #print(i0_levels[ts])
215     decomp = stl(i0_levels[[ts]], s.window='periodic') #additive seasonality seems right from
        graphs
216     graph = autoplot(decomp) + labs(title = names(i0_levels)[ts])
217     deseason = i0_levels[[ts]] - decomp$time.series[, 'seasonal']
218     decompositions[[length(decompositions)+1]] = decomp
219     dec_graphs[[length(dec_graphs)+1]] = graph
220     deseasonalized_ts[[length(deseasonalized_ts)+1]] = deseason
221 }
222 names(deseasonalized_ts) = names(i0_levels) #name the series in deseasonalized_ts
223 names(decompositions) = names(i0_levels) #name the series in decompositions
224
225 rm(graph, decomp, deseason)
226
227 #Generate and export plots with the decomposition graphs
228 decom_i = grid.arrange(dec_graphs[[1]], dec_graphs[[2]], ncol=2)
229 # ggsave('IMAGES/decomposition_i.png', plot=decom_i, width = 12, height = 8)
230
231 ### Run OLS regression on each component of the TS
232 #create a df with all the series and the monthly dummies
233 start_date = as.Date("1990-01-01")
234 end_date = as.Date("2022-12-01") # Assuming December 2022
235 all_dates = seq(start_date, end_date, by = "month")
236
237 df_i0level = data.frame(Date = all_dates)
238 for (i in seq_along(i0_levels)) {
239     col_name = paste0("Series_", i)

```

```

240   df_i0level[[col_name]] = i0_levels[[i]]
241 }
242 colnames(df_i0level) = c('Date', names(i0_levels))
243 rownames(df_i0level) = df_i0level$Date #date as index
244 df_i0level$MONTH=month(df_i0level$Date) #Get month dummies
245 df_i0level[paste0("M", 1:12)] = as.data.frame(t(sapply(df_i0level$MONTH, tabulate, 12)))
246
247 df_i0level$Date = NULL #no more need date column
248 df_i0level$MONTH = NULL #no more need MONTH column
249
250 df_i0level$trend = seq_along(df_i0level$infl_e) #deterministic trend
251
252 # Run OLS regression
253 infl_e_ols_dec = lm(infl_e ~ trend + M1 + M2 + M3 + M4 + M5 + M6 + M7 + M8 + M9 + M10 + M11,
254   data = df_i0level)
255
256 deflator_ols_dec = lm(deflator ~ trend + M1 + M2 + M3 + M4 + M5 + M6 + M7 + M8 + M9 + M10 +
257   M11, data = df_i0level)
258
259 stargazer(infl_e_ols_dec, deflator_ols_dec, type='text')
260
261 # stargazer(infl_e_ols_dec, deflator_ols_dec,
262 #   type='latex', out="TABLES/ols_decomp_levels", label='tab:ols_dec_levels',
263 #   title='OLS decomposition of I(0) series' )
264
265 ## remove drifts and trend and store in new list
266 i0_levels_no_drift = list()
267
268 ct_infl = coef(infl_e_ols_dec)[1]
269 trend_infl = coef(infl_e_ols_dec)[2]
270 m5_infl = coef(infl_e_ols_dec)[7]
271 i0_levels_no_drift$infl_e = i0_levels$infl_e - ct_infl - (df_i0level$trend*trend_infl) - (df_
272   i0level$M5*m5_infl)
273
274 ct_defl = coef(deflator_ols_dec)[1]
275 trend_defl = coef(deflator_ols_dec)[2]
276 i0_levels_no_drift$deflator = i0_levels$deflator - ct_defl - (df_i0level$trend*trend_defl)
277
278 ### Find p and q with PACF and ACF
279
280 acf_infl = ggAcf(i0_levels_no_drift$infl_e, lag.max= 24) + labs(title = 'ACF - infl_e')
281 acf_defl= ggAcf(i0_levels_no_drift$deflator, lag.max= 24) + labs(title = 'ACF - deflator')
282
283 ggsave('IMAGES/acf_infl.png', plot=acf_infl, width = 12, height = 8)
284 ggsave('IMAGES/acf_defl.png', plot=acf_defl, width = 12, height = 8)
285
286 pacf_infl = ggPacf(i0_levels_no_drift$infl_e, lag.max= 24)+ labs(title = 'PACF - infl_e')
287 pacf_defl = ggPacf(i0_levels_no_drift$deflator, lag.max= 24) + labs(title = 'PACF - deflator')

```

```

285
286 ggsave('IMAGES/pacf_infl.png', plot= pacf_infl, width = 12, height=8)
287 ggsave('IMAGES/pacf_defl.png', plot= pacf_defl, width = 12, height=8)
288
289
290 ### Minimize information criteria
291 #inflation expectation
292 arma_bic_infl = critMatrix(i0_levels_no_drift$infl_e, p.max = 4, q.max = 15, criterion='bic')
293 stargazer(arma_bic_infl, type='text', flip=T)
294 # stargazer(arma_bic_infl, type='latex', flip=T,
295 #           out= 'TABLES/BIC_arma_infl.tex', label="tab:bic_infl",
296 #           title= "Information criteria on the parameters of ARMA for infl-e")
297
298 #deflator (too slow, I commented the line to avoid breaking everything)
299 arma_bic_defl = critMatrix(i0_levels_no_drift$deflator, p.max = 5, q.max = 18, criterion='bic'
300 )
301 stargazer(arma_bic_defl, type='text', flip=T)
302 # stargazer(arma_bic_defl, type='latex', flip=T,
303 #           out= 'TABLES/BIC_arma_deflator.tex', label="tab:bic_deflator",
304 #           title= "Information criteria on the parameters of ARMA for GDP deflator")
305
306 ## fit ARMA model
307 arma_infl = arima(i0_levels_no_drift$infl_e, order= c(1,0,0))
308 stargazer(arma_infl, type='text')
309
310 arma_defl = arima(i0_levels_no_drift$infl_e, order= c(1,0,0))
311 stargazer(arma_defl, type='text')
312
313
314 #####
315 #3. Decompositions in deltas
316 #####
317 # Same idea for series in deltas
318
319 decompositions_d = list() #store the decompositions
320 dec_graphs_d= list() #store the decomposition graphs
321 deseasonalized_ts_d = list() #store the deseasonalized ts, if seasonality ends up being
322     important
323 for (ts in 1:length(deltas)) {
324     #print(deltas[ts])}
325     decomp = stl(deltas[[ts]], s.window='periodic') #additive seasonality seems right from
326     graphs
327     graph = autoplot(decomp) + labs(title = names(deltas)[ts])
328     deseason = deltas[[ts]] - decomp$time.series[, 'seasonal']
329     decompositions_d[[length(decompositions_d)+1]] = decomp
330     dec_graphs_d[[length(dec_graphs_d)+1]] = graph
331     deseasonalized_ts_d[[length(deseasonalized_ts_d)+1]] = deseason

```

```

330 }
331 names(deseasonalized_ts_d) = names(deltas) #name the series in deseasonalized_ts
332 names(decompositions_d) = names(deltas) #name the series in decompositions
333 rm(graph, decomp, deseason)
334
335 #Generate and export plots with the decomposition graphs
336 decom_ii = grid.arrange(dec_graphs_d[[1]],dec_graphs_d[[2]], dec_graphs_d[[3]],dec_graphs_d
  [[4]], ncol=2)
337 # ggsave('IMAGES/decomposition_ii.png', plot=decom_ii, width = 12, height = 16)
338
339 ##Create single image with all decompositions, looks better on latex
340 all_dec = grid.arrange(dec_graphs[[1]], dec_graphs[[2]], dec_graphs_d[[1]],dec_graphs_d[[2]],
  dec_graphs_d[[3]],dec_graphs_d[[4]], ncol=3)
341 # ggsave('IMAGES/all_decompositions.png', plot=all_dec, width = 12, height = 16)
342
343
344
345 ### Run OLS regression on each component of the TS
346 #create a df with all the series and the monthly dummies
347 start_date = as.Date("1990-02-01")
348 end_date = as.Date("2022-12-01")
349 all_dates = seq(start_date, end_date, by = "month")
350
351 df_deltas = data.frame(Date = all_dates)
352 for (i in seq_along(deltas)) {
353   col_name = paste0("Series_", i)
354   df_deltas[[col_name]] = deltas[[i]]
355 }
356 colnames(df_deltas) = c('Date', names(deltas))
357 rownames(df_deltas) = df_deltas$Date #date as index
358 df_deltas$MONTH=month(df_deltas$Date) #Get month dummies
359 df_deltas[paste0("M", 1:12)] = as.data.frame(t(apply(df_deltas$MONTH, tabulate, 12)))
360
361 df_deltas$Date = NULL #no more need date column
362 df_deltas$MONTH = NULL #no more need MONTH column
363
364 df_deltas$trend = seq_along(df_deltas$d_gdp) #deterministic trend
365
366 # Run OLS regression
367 d_gdp_ols_dec = lm(d_gdp ~ trend + M1 + M2 + M3 + M4 + M5 + M6 + M7 + M8 + M9 + M10 + M11,
  data = df_deltas)
368 d_dpi_ols_dec = lm(d_dpi ~ trend + M1 + M2 + M3 + M4 + M5 + M6 + M7 + M8 + M9 + M10 + M11,
  data = df_deltas)
369 d_rate_ols_dec = lm(d_rate ~ trend + M1 + M2 + M3 + M4 + M5 + M6 + M7 + M8 + M9 + M10 + M11,
  data = df_deltas)
370 d_splong_ols_dec = lm(d_splong ~ trend + M1 + M2 + M3 + M4 + M5 + M6 + M7 + M8 + M9 + M10 +
  M11, data = df_deltas)
371

```

```

372 stargazer(d_gdp_ols_dec,d_dpi_ols_dec,d_rate_ols_dec, d_splong_ols_dec, type='text')
373 # stargazer(d_gdp_ols_dec,d_dpi_ols_dec,d_rate_ols_dec, d_splong_ols_dec,
374 #           type='latex', out="TABLES/ols_decomp_deltas", label='tab:ols_dec_deltas',
375 #           title='OLS decomposition of the first difference of I(1) series' )
376
377 ## remove drifts and store in new list (remove trend for rate only)
378 deltas_no_drift_trend = list()
379
380 ct_gdp = coef(d_gdp_ols_dec)[1]
381 deltas_no_drift_trend$d_gdp = deltas$d_gdp - ct_gdp
382
383 ct_dpi = coef(d_dpi_ols_dec)[1]
384 deltas_no_drift_trend$d_dpi = deltas$d_dpi - ct_dpi
385
386 ct_rate = coef(d_rate_ols_dec)[1]
387 trend_rate = coef(d_rate_ols_dec)[2]
388 m6_rate = coef(d_rate_ols_dec)[8]
389 deltas_no_drift_trend$d_rate = deltas$d_rate - ct_rate - (df_deltas$trend * trend_rate) - (m6_
    rate*df_deltas$M6)
390
391 ct_sp = coef(d_splong_ols_dec)[1]
392 m9_sp = coef(d_splong_ols_dec)[11]
393 m10_sp = coef(d_splong_ols_dec)[12]
394 deltas_no_drift_trend$d_splong = deltas$d_splong - ct_sp - (m9_sp*df_deltas$M9) - (m10_sp*df_
    deltas$M10)
395
396
397 ### Find p and q with PACF and ACF
398
399 acf_gdp = ggAcf(deltas_no_drift_trend$d_gdp, lag.max= 24) + labs(title = 'ACF - d_gdp')
400 acf_dpi = ggAcf(deltas_no_drift_trend$d_dpi, lag.max= 24) + labs(title = 'ACF - d_dpi')
401 acf_rate = ggAcf(deltas_no_drift_trend$d_rate, lag.max= 24) + labs(title = 'ACF - d_rate')
402 acf_sp = ggAcf(deltas_no_drift_trend$d_splong, lag.max= 24) + labs(title = 'ACF - d_splong')
403
404 ggsave('IMAGES/acf_gdp.png', plot=acf_gdp, width = 12, height = 8)
405 ggsave('IMAGES/acf_dpi.png', plot=acf_dpi, width = 12, height = 8)
406 ggsave('IMAGES/acf_rate.png', plot=acf_rate, width = 12, height = 8)
407 ggsave('IMAGES/acf_sp.png', plot=acf_sp, width = 12, height = 8)
408
409 pacf_gdp = ggPacf(deltas_no_drift_trend$d_gdp, lag.max= 24)+ labs(title = 'PACF - d_gdp')
410 pacf_dpi = ggPacf(deltas_no_drift_trend$d_dpi, lag.max= 24)+ labs(title = 'PACF - d_dpi')
411 pacf_rate = ggPacf(deltas_no_drift_trend$d_rate, lag.max= 24)+ labs(title = 'PACF - d_rate')
412 pacf_sp = ggPacf(deltas_no_drift_trend$d_splong, lag.max= 24)+ labs(title = 'PACF - d_splong')
413
414 ggsave('IMAGES/pacf_gdp.png', plot= pacf_gdp, width = 12, height=8)
415 ggsave('IMAGES/pacf_dpi.png', plot= pacf_dpi, width = 12, height=8)
416 ggsave('IMAGES/pacf_rate.png', plot= pacf_rate, width = 12, height=8)
417 ggsave('IMAGES/pacf_sp.png', plot= pacf_sp, width = 12, height=8)

```

```

418
419
420 # minimize information criteria
421 #gdp
422 arma_bic_gdp = critMatrix(deltas_no_drift_trend$d_gdp, p.max = 13, q.max = 2, criterion='bic')
423 stargazer(arma_bic_gdp, type='text', flip=F)
424 # stargazer(arma_bic_gdp, type='latex', flip=F,
425 #           out= 'TABLES/BIC_arma_gdp.tex', label="tab:bic_gdp",
426 #           title= "Information criteria on the parameters of ARMA for d-gdp")
427
428
429 #dpi
430 arma_bic_dpi = critMatrix(deltas_no_drift_trend$d_dpi, p.max = 12, q.max = 12, criterion='bic'
431 )
432 stargazer(arma_bic_dpi, type='text', flip=F)
433 # stargazer(arma_bic_dpi, type='latex', flip=F,
434 #           out= 'TABLES/BIC_arma_dpi.tex', label="tab:bic_dpi",
435 #           title= "Information criteria on the parameters of ARMA for d-dpi")
436
437
438 #rate
439 arma_bic_rate = critMatrix(deltas_no_drift_trend$d_rate, p.max = 6, q.max = 10, criterion='bic
440 ')
441 stargazer(arma_bic_rate, type='text', flip=T)
442 # stargazer(arma_bic_rate, type='latex', flip=T,
443 #           out= 'TABLES/BIC_arma_rate.tex', label="tab:bic_rate",
444 #           title= "Information criteria on the parameters of ARMA for d-rate")
445
446
447 #dpi
448 arma_bic_sp = critMatrix(deltas_no_drift_trend$d_splong, p.max =6, q.max = 1, criterion='bic')
449 stargazer(arma_bic_sp, type='text', flip=F)
450 # stargazer(arma_bic_sp, type='latex', flip=F,
451 #           out= 'TABLES/BIC_arma_sp.tex', label="tab:bic_sp",
452 #           title= "Information criteria on the parameters of ARMA for d-splong")
453
454
455 ## COINTEGRATION
456 levels_var = data.frame(
457   dpi = allts$dpi,
458   gdp = allts$gdp,
459   rate = allts$rate,
460   splong = allts$splong)
461
462 #levels_var$timestamp = time(deseasonalized_ts$deflator)
463 names_level = c('splong', 'rate', 'gdp', 'dpi')

```

```

464
465 lag_order = VARselect(levels_var)
466 res = lag_order$criteria
467
468
469 johansen_test = ca.jo(levels_var, type='trace', ecdet='trend', K=10)
470 summary(johansen_test)
471 # r is rank of matrix == number of cointegration relationship.
472 #no cointegration
473 johansen_table = xtable(johansensum)
474 print(latex_table, file=file_path)
475
476
477
478 test = data.frame(
479     i= allts$dpi,
480     ii = allts$spplong,
481     iii = allts$gdp)
482
483 VARselect(test)$selection
484 summary(ca.jo(test, type='trace', ecdet='trend', K=20))

```