Advanced Econometrics Project

22 04 2024

Introduction

This analysis explores the relationship between economic efficiency, measured by the Capacity Utilization Rate, and inflation, gauged by the CPI Inflation Rate, using time-series data from FRED. The study aims to understand their interplay and the broader economic implications by examining the data's stationarity, cointegration, and causality. Stationarity is checked using ADF, PP, and KPSS tests, while cointegration is investigated through EngleGranger and Johansen tests. ARDL, VAR, and VEC models assess short-run and long-run effects, with causality examined via Granger-causality and Toda-Yamamoto tests. Results from R code illustrate the intricate connections between these indicators and economic health.

```r
# Required libraries for econometric analysis

library(dynamac)          # Dynamac for ARDL models library(forecast)
# Forecasting functions for ARIMA models

## Registered S3 method overwritten by 'quantmod':
##   method            from ##
as.zoo.data.frame zoo

library(tseries)          # Time series testing and modeling library(nlme)
# Nonlinear and linear mixed effects models

##
## Attaching package: 'nlme'

## The following object is masked from 'package:forecast':
##
##     getResponse

library(pdfetch)          # Fetching data from online databases
library(zoo)              # S3 infrastructure for regular and irregular time
series

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(urca)             # Unit root and cointegration tests library(vars)
# VAR, SVAR and SVEC models

## Loading required package: MASS

## Loading required package: strucchange

## Loading required package: sandwich ## Loading required package: lmtest

library(car)              # Companion to Applied Regression, for diagnostics

## Loading required package: carData

library(dynlm)            # Dynamic linear models and time series regression
library(tsDyn)            # Nonlinear time series models with regime
switching library(gets)             # General-to-Specific (GETS) modeling ##
Loading required package: parallel

##
## Attaching package: 'gets'

## The following object is masked from 'package:car':
##
##     logit

library(readxl)           # Reading Excel files
library(aod)              # Analysis of Overdispersed Data, for Wald tests
library(egcm)             # Engle-Granger cointegration models

## Loading required package: xts library(aTSA)

# Advanced Time Series Analysis

##
## Attaching package: 'aTSA'

## The following object is masked from 'package:vars':
##
##     arch.test

## The following objects are masked from 'package:tseries':
##
##     adf.test, kpss.test, pp.test

## The following object is masked from 'package:forecast':
##
##     forecast
```

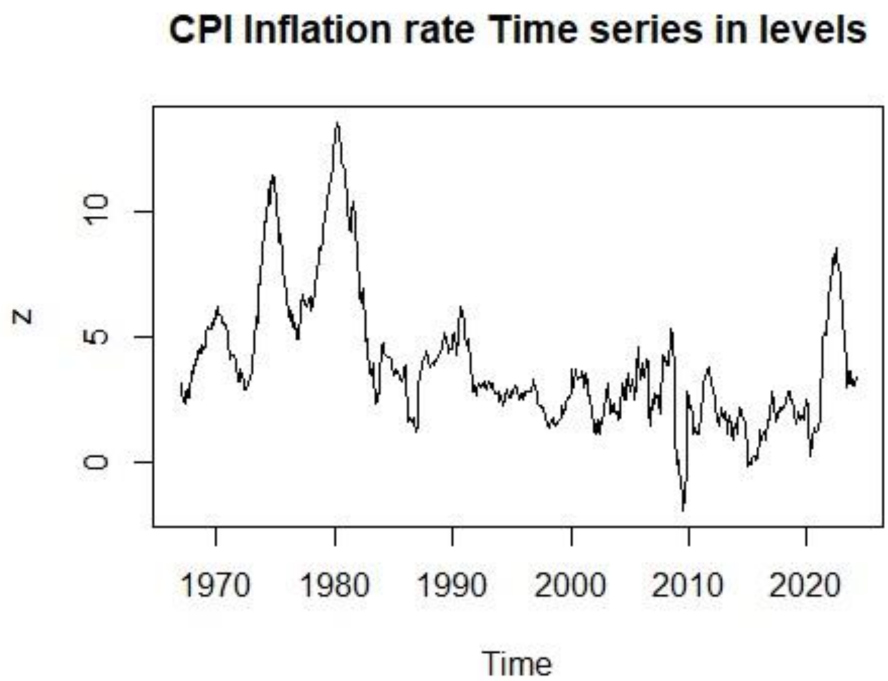
```
## The following object is masked from 'package:graphics':
##
##     identify

# Fetching monthly data for CPI and TCU from the FRED database

# Consumer Price Index (CPI) retrieval and setup CPI
= pdfetch_FRED("CPIAUCSL")  # Acquire CPI data
names(CPI) = "CPI"  # Set the series name to 'CPI'

# Calculate the annual inflation rate based on CPI changes
Inflation = diff(log(CPI), lag = 12) * 100  # Calculate year-on-year log
differences and convert to percentage names(Inflation) = "Inflation"  #
Rename the series to 'Inflation'

# Set the time series object for inflation starting from 1947, with monthly
observations
Inflation = ts(Inflation, start=c(1947, 1), frequency=12)
Inflation = na.omit(Inflation)  # Remove missing values from the series,
starting in January 1948

# Obtain and format the Total Capacity Utilization (TCU) data, indexed as a
percentage of capacity, seasonally adjusted TCU = pdfetch_FRED("TCU")  #
Retrieve TCU data names(TCU) = "TCU"  # Label the series as 'TCU'

# Define the time series object for TCU beginning from 1967, with monthly
frequency
TCU = ts(TCU, start=c(1967, 1), frequency=12)

 # Creating the Dataset with the downloaded Inflation and TCU
data.set = na.omit(            ts.intersect(

       Inflation,
       TCU,

           dframe=TRUE))




Inflation   = ts( data.set$Inflation,    start=c(1967, 1),  frequency=12)
TCU         = ts( data.set$TCU,          start=c(1967, 1),  frequency=12)
```

Unit root tests (ADF, PP, KPSS) assess stationarity in Inflation and TCU, revealing trends and predictability.

CPI Inflation

Graphic 1 (Levels) :

```
z= Inflation
plot(z, main = "CPI Inflation rate Time series in levels")
```



CPI Unit Root Tests (Levels):

Setting Max Lags:

We set the maximum lags for unit root tests on series 'z' using trunc() to ensure integer values. The I operator forwards results.

```
max.lags = (12*((length(z)/100) ^(1/4))) |> trunc()     max.lags

## [1] 19
```

The formula calculates 19 lags for unit root tests on series 'z', accounting for autocorrelation and ensuring accurate stationarity assessments.

ADF:

The Augmented Dickey-Fuller test checks if a series, like CPI Inflation, is non-stationary with a unit root or not, using lagged differences in the regression.

```
# Performing Augmented Dickey-Fuller (ADF) Tests on the selected time series:
      # ADF Test: Null hypothesis (Ho) = presence of a unit root = non-
```

```
stationarity at level
# Alternative hypothesis (H1) = absence of a unit root = stationarity

    ur.df( z,   type="drift"   , selectlags="BIC",   lags=max.lags ) |>
summary()

##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #  ##
###############################################  ##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag) ##
## Residuals:
##      Min       1Q   Median       3Q      Max  ##
-1.66986 -0.15427  0.00089  0.16725  1.61693
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.054761   0.021520   2.545 0.011169 *
## z.lag.1      -0.014044   0.004632  -3.032 0.002527 **
## z.diff.lag1   0.462898   0.038603  11.991  < 2e-16 ***
## z.diff.lag2  -0.014136   0.042580  -0.332 0.739999
## z.diff.lag3   0.048581   0.042224   1.151 0.250346
## z.diff.lag4   0.034795   0.036170   0.962 0.336416
## z.diff.lag5   0.023031   0.035833   0.643 0.520621
## z.diff.lag6  -0.016730   0.035774  -0.468 0.640190
## z.diff.lag7   0.095699   0.035736   2.678 0.007594 **
## z.diff.lag8  -0.021600   0.035903  -0.602 0.547633
## z.diff.lag9   0.042267   0.035850   1.179 0.238828
## z.diff.lag10  0.081681   0.036010   2.268 0.023638 *
## z.diff.lag11  0.139585   0.036191   3.857 0.000126 ***
## z.diff.lag12 -0.550240   0.036685 -14.999  < 2e-16 ***
## z.diff.lag13  0.156050   0.042705   3.654 0.000279 ***
## z.diff.lag14 -0.018835   0.043127  -0.437 0.662453    ##
z.diff.lag15  0.139461   0.039288   3.550 0.000414 *** ##
---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.2955 on 650 degrees of freedom
## Multiple R-squared:  0.4295, Adjusted R-squared:  0.4154
## F-statistic: 30.58 on 16 and 650 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -3.0319 4.5971  ##
```

```
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.43 -2.86 -2.57
## phi1  6.43  4.59  3.78
```

The ADF test suggests 'z' is non-stationary, with significant autocorrelation at lag 1, and a test statistic of -3.0319 implies non-stationarity.

PP:

The Phillips-Peron test checks for unit roots without needing lags, contrasting with the ADF's approach to account for serial correlation.

```
 # Executing Phillips-Perron (PP) Unit Root Tests:
      # PP Test: Null hypothesis (Ho) posits non-stationarity due to a unit
root
      ur.pp( z, type="Z-tau"  ,  model="constant" , lags="long" ) |>
summary()    # utilizing "urca" package

##
## ##################################
## # Phillips-Perron Unit Root Test #  ##
##################################
##
## Test regression with intercept
##
##
## Call:
## lm(formula = y ~ y.l1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max  ##
-2.57226 -0.20950 -0.01202  0.20297  2.07847
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.039975   0.025773   1.551    0.121     ##
y.l1        0.989916   0.005407 183.075   <2e-16 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.3829 on 684 degrees of freedom
## Multiple R-squared:   0.98,  Adjusted R-squared:   0.98
## F-statistic: 3.352e+04 on 1 and 684 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic, type: Z-tau  is: -2.8753  ##
##          aux. Z statistics ##
Z-tau-mu            2.3779 ##
```

```
## Critical values for Z statistics:
##                      1pct      5pct     10pct
## critical values -3.442307 -2.866109 -2.569206
```

Phillips-Perron test suggests non-stationarity in time series with persistent shocks, indicated by statistic -2.8753.

KPSS:

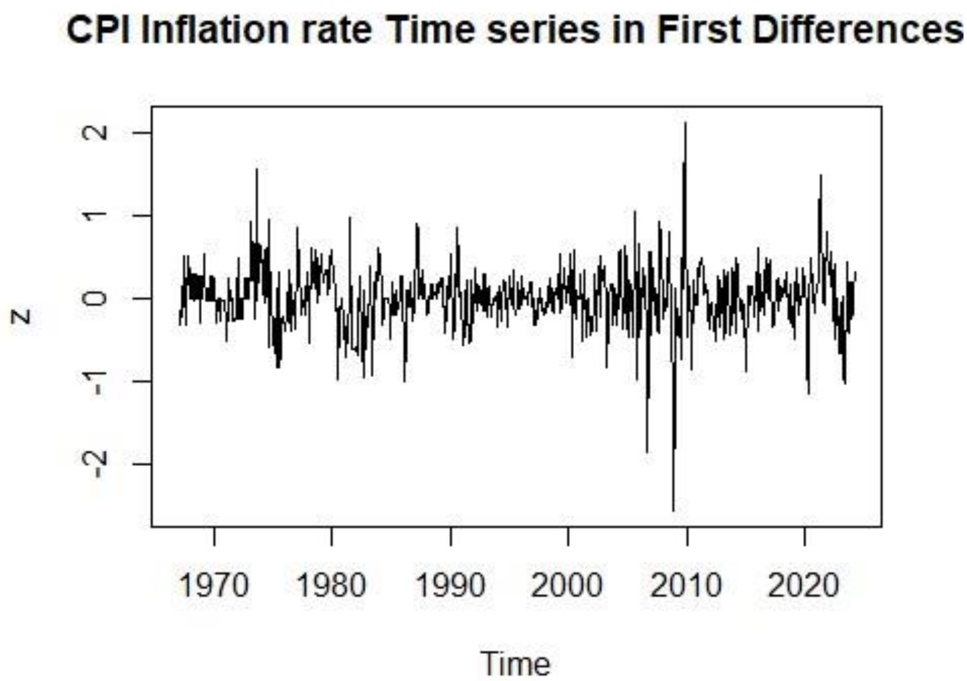
The KPSS test, employed to assess a time series' stationarity, assumes a contrasting null hypothesis, positing the absence of a unit root and the series' stationarity. It offers various configurations for execution.

```r
# Executing KPSS Unit Root Tests on selected time series:
      # KPSS Test: Null hypothesis (Ho) asserts the series is stationary
[inverse of ADF/PP]
      # "mu" represents testing around a constant without trend

      ur.kpss( z, type="mu"  ,  lags="long" ) |> summary()     # utilizing
"urca" package
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 19 lags.
##
## Value of test-statistic is: 1.3775  ##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

PSS statistic of 1.3781 exceeds 1% critical value (0.739), indicating a non-stationary series.

```r
z = diff(Inflation)
plot(z, main = "CPI Inflation rate Time series in First Differences")
```



ADF (First Differences):

```
# ADF Test: Null Hypothesis (Ho) - Presence of a unit root implies
nonstationarity at levels
# Alternative Hypothesis (H1) - Absence of a unit root indicates stationarity
ur.df( z, type="drift", selectlags="BIC", lags=max.lags ) |> summary()

##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #  ##
###############################################  ##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag) ##
## Residuals:
##      Min       1Q   Median       3Q      Max  ##
-1.75890 -0.15854  0.00385  0.16671  1.57577
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.806e-06  1.162e-02   0.001 0.999327
## z.lag.1      -5.963e-01  7.293e-02  -8.176 1.54e-15 ***
## z.diff.lag1   6.664e-02  6.831e-02   0.976 0.329639
```

```
## z.diff.lag2   4.503e-02  6.796e-02   0.663 0.507779
## z.diff.lag3   1.811e-02  6.667e-02   0.272 0.786038
## z.diff.lag4   7.020e-02  6.484e-02   1.083 0.279402
## z.diff.lag5   9.790e-02  6.254e-02   1.565 0.117996
## z.diff.lag6   8.030e-02  6.033e-02   1.331 0.183648
## z.diff.lag7   1.705e-01  5.696e-02   2.993 0.002867 **
## z.diff.lag8   1.513e-01  5.399e-02   2.802 0.005222 **
## z.diff.lag9   1.851e-01  5.051e-02   3.665 0.000267 ***
## z.diff.lag10  2.646e-01  4.548e-02   5.818 9.34e-09 ***
##  z.diff.lag11   4.004e-01   4.145e-02     9.660    <  2e-16  *** ##
z.diff.lag12 -1.688e-01  3.918e-02  -4.308 1.90e-05 *** ## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.2997 on 652 degrees of freedom
## Multiple R-squared:  0.5049, Adjusted R-squared:  0.4951  ## F-
statistic: 51.15 on 13 and 652 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -8.1757 33.4218  ##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.43 -2.86 -2.57
## phi1  6.43  4.59  3.78
```

The ADF test shows a statistic of -8.1757, much lower than the 1% critical value of -3.43, indicating the series 'z' is likely stationary and lacks a unit root.

PP (First Differences):

```
# Executing Phillips-Perron Test:
# PP Null Hypothesis: The series contains a unit root, indicating
nonstationarity ur.pp( z, type="Z-tau", model="constant", lags="long"
) |> summary()

##
## ##################################
## # Phillips-Perron Unit Root Test #  ##
##################################
##
## Test regression with intercept
##
##
## Call:
## lm(formula = y ~ y.l1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max  ##
-2.09957 -0.17593  0.00663  0.19714  1.65255  ##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0008785  0.0134337   0.065    0.948     ## y.l1
0.4016808  0.0350383  11.464   <2e-16 *** ## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.3516 on 683 degrees of freedom
## Multiple R-squared:  0.1614, Adjusted R-squared:  0.1601
## F-statistic: 131.4 on 1 and 683 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic, type: Z-tau  is: -17.2184  ##
##         aux. Z statistics ## Z-tau-mu
0.0652
##
## Critical values for Z statistics:
##                       1pct      5pct     10pct
## critical values -3.44232 -2.866115 -2.569209
```

The Phillips-Perron test yields a test statistic of -17.2184, which is significantly lower than critical thresholds, suggesting the time series is stationary and lacks a persistent unit root trend.

KPSS (First Differences):

```
 # KPSS Test: Null Hypothesis asserts no unit root exists, in contrast to
ADF/PP ur.kpss( z, type="mu", lags="long" ) |>
summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 19 lags.
##
## Value of test-statistic is: 0.0339  ##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

The KPSS test statistic of 0.0339, below the 1% critical value of 0.739, suggests the time series is stationary, indicating no unit root and consistent mean over time.

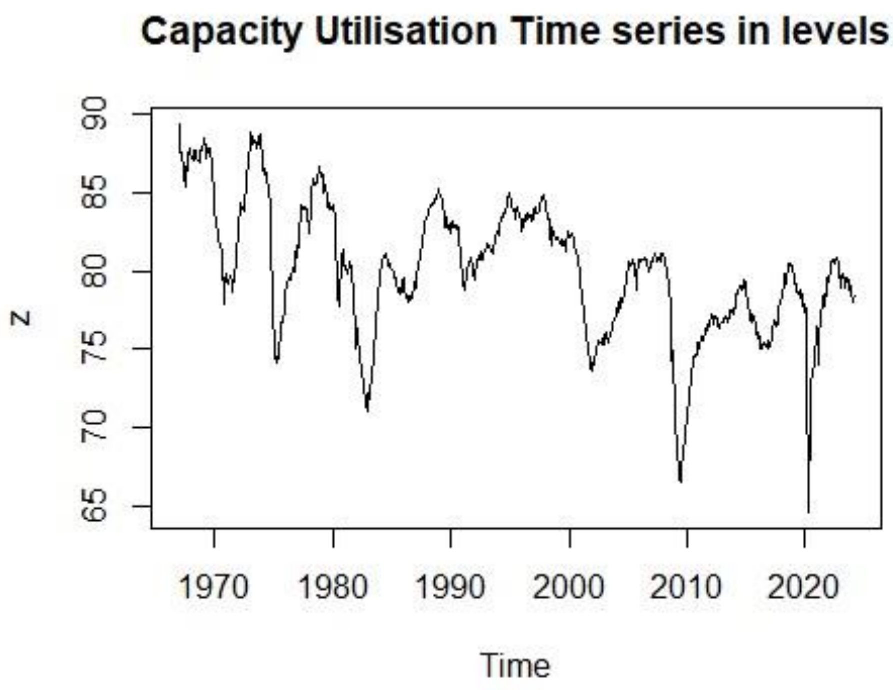
TCU:

Unit Root Tests

Graphic 2 (Levels):

```
z = TCU
```

```r
plot(z, main = "Capacity Utilisation Time series in levels")
```



ADF (Levels):

```r
# ADF Test Commentary:
# ADF Null Hypothesis (Ho): The series contains a unit root, indicating it
follows an I(1) process
# ADF Null Hypothesis (Ho): The series contains a unit root, meaning
it's non-stationary at level ur.df( z, type="drift", selectlags="BIC",
lags=max.lags ) |> summary()

##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #  ##
###############################################  ##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max  ##
-9.1842 -0.2989  0.0349  0.3203  3.6852
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.812823   0.535631   3.384 0.000755 ***
## z.lag.1     -0.022810   0.006697  -3.406 0.000699 *** ##
z.diff.lag   0.294379   0.036997   7.957 7.63e-15 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.7011 on 664 degrees of freedom
## Multiple R-squared:  0.09675,    Adjusted R-squared:  0.09403
## F-statistic: 35.56 on 2 and 664 DF,  p-value: 2.126e-15
##
##
## Value of test-statistic is: -3.406 5.8576  ##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.43 -2.86 -2.57
## phi1  6.43  4.59  3.78
```

ADF results indicate a significant intercept and lag, hinting at mean reversion in 'z'. A test statistic more negative than critical values suggests stationarity.

PP (Levels):

```
 # PP:  Ho = series has a unit root = series is non-stationary    ur.pp( z,
type="Z-tau"  ,  model="constant" , lags="long" ) |> summary()
```

```
##
## ##################################
## # Phillips-Perron Unit Root Test #  ##
##################################
##
## Test regression with intercept
##
##
## Call:
## lm(formula = y ~ y.l1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max  ##
-10.0524  -0.2931   0.0583   0.3412   4.0611
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.49745    0.53483     2.8  0.00526 **
```

```
## y.l1         0.98110    0.00667   147.1  < 2e-16 *** ## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.7311 on 684 degrees of freedom
## Multiple R-squared:  0.9694, Adjusted R-squared:  0.9693
## F-statistic: 2.163e+04 on 1 and 684 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic, type: Z-tau  is: -3.7552  ##
##          aux. Z statistics ## Z-tau-mu
3.7299
##
## Critical values for Z statistics:
##                      1pct      5pct     10pct
## critical values -3.442307 -2.866109 -2.569206
```

With a calculated value of -3.7506 surpassing the critical threshold of -3.442307, we reject the non-stationarity null hypothesis, implying that the time series exhibits stationarity.

KPSS (Levels):

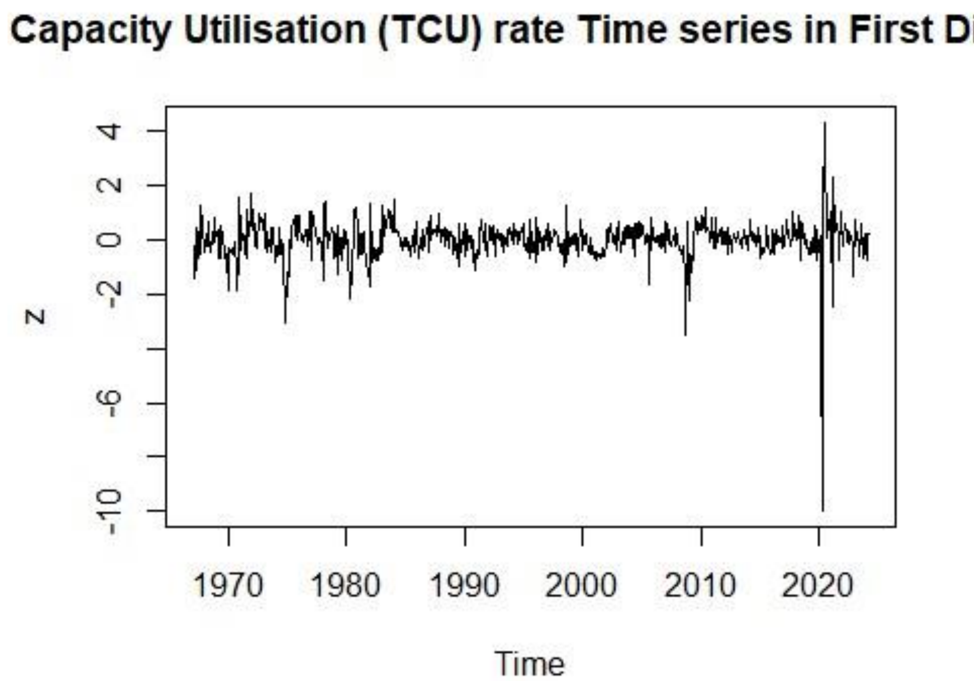
```
# KPSS:   Ho = series does not have a unit root [opposite of ADF and PP]
ur.kpss( z, type="mu"   ,  lags="long" ) |> summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 19 lags.
##
## Value of test-statistic is: 1.3245  ##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

With a KPSS test statistic of 1.3219, exceeding the 1% critical value of 0.739, we refute the null hypothesis, suggesting the series is non-stationary and may possess a unit root

Graphic 2A (First Differences):

```
z = diff(TCU)
      plot(z, main = "Total Capacity Utilisation (TCU) rate Time series in
First Differences")
```



ADF (First Differences) for TCU:

```
# ADF Analysis: Null Hypothesis posits that the series is an I(1)
integrated process ur.df( z, type="drift", selectlags="BIC", lags=max.lags )
|> summary()

##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #  ##
###############################################  ##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag) ##
## Residuals:
##     Min      1Q  Median      3Q     Max  ##
-9.0556 -0.3106  0.0085  0.3090  3.9156
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.00958    0.02741  -0.349    0.727
## z.lag.1     -0.74027    0.04644 -15.942   <2e-16 *** ##
z.diff.lag   0.03473    0.03882   0.895    0.371
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.7073 on 663 degrees of freedom
## Multiple R-squared:  0.3584, Adjusted R-squared:  0.3565  ##
F-statistic: 185.2 on 2 and 663 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -15.9417 127.0698  ##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.43 -2.86 -2.57
## phi1  6.43  4.59  3.78
```

The ADF test result, with a t-value of -15.9417, suggests significant evidence against a unit root, pointing to the likely stationarity of the time series and consistent statistical properties over time.

PP (First Differences) for TCU:

```
# Phillips-Perron Test: Null Hypothesis assumes a unit root, implying
nonstationarity in the series ur.pp( z, type="Z-tau", model="constant",
lags="long" ) |> summary()
```

```
##
## ##################################
## # Phillips-Perron Unit Root Test #  ##
##################################
##
## Test regression with intercept
##
##
## Call:
## lm(formula = y ~ y.l1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max  ##
-9.1124 -0.3086  0.0111  0.3043  4.0279
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.009422   0.026942  -0.350    0.727    ##
y.l1         0.277770   0.036655   7.578 1.15e-13 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.705 on 683 degrees of freedom
## Multiple R-squared:  0.07756,    Adjusted R-squared:  0.07621
## F-statistic: 57.42 on 1 and 683 DF,  p-value: 1.147e-13
##
```

```
##
## Value of test-statistic, type: Z-tau  is: -20.3004 ##
##          aux. Z statistics ##
Z-tau-mu           -0.3634
##
## Critical values for Z statistics:
##                     1pct      5pct     10pct
## critical values -3.44232 -2.866115 -2.569209
```

The PP test's Z-tau statistic of -20.3004, well below the 1% critical value, robustly rejects the unit root hypothesis, indicating the series' stationarity and absence of persistent trends.

KPSS (First Differences)for TCU:

```
 # KPSS Analysis: Testing for stationarity, with the Null Hypothesis
assuming no unit root (contrary to ADF/PP) ur.kpss( z, type="mu", lags="long"
) |> summary()
```

```
##
## #######################
## # KPSS Unit Root Test #
## #######################
##
## Test is of type: mu with 19 lags.
##
## Value of test-statistic is: 0.0362  ##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

KPSS statistic 0.0355, below the 1% critical threshold of 0.739, indicates differenced series stationarity, contradicting ADF/PP tests.

Analyzing the co-integration of the Consumer Price Index (CPI) Inflation Rate with Total Capacity Utilization (TCU):

Co-integration indicates a long-term equilibrium among non-stationary time series. Despite individual randomness, co-integrated variables share a stable, linear relationship.

There will be two tests for cointegration conducted:

Engle-Granger Cointegration test:

The Engle-Granger test evaluates if non-stationary time series are co-integrated, suggesting they maintain a long-term relationship despite individual non-stationarity.

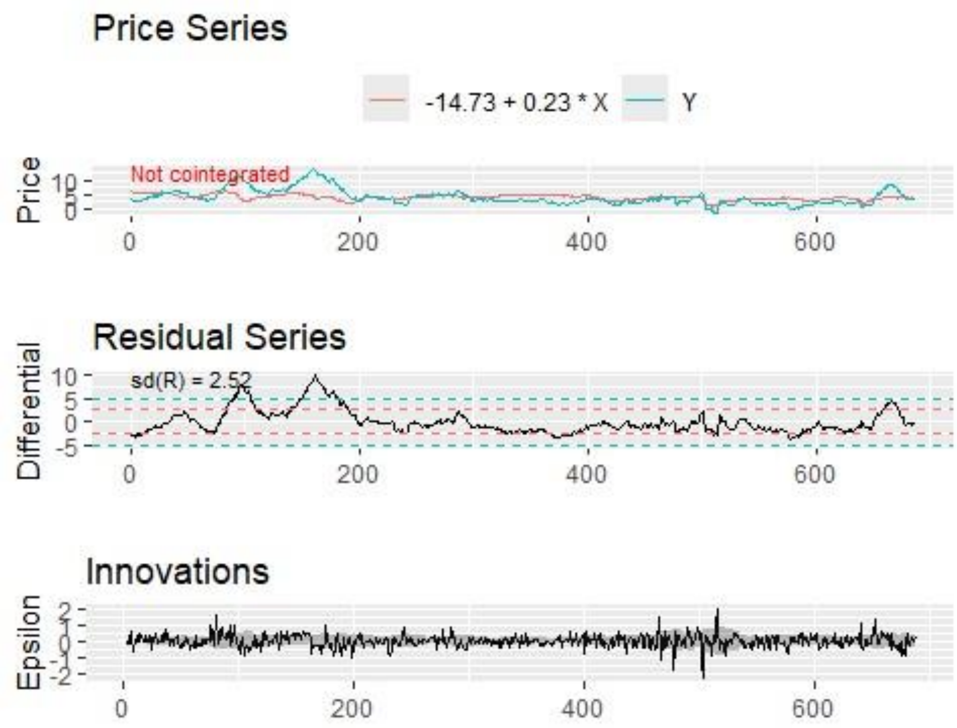
```
z = Inflation #Using
egcm package
```

```
EGCM = egcm( Y= Inflation, X= TCU, include.const   = TRUE)  EGCM

## Y[i] =   0.2329 X[i] -  14.7253 + R[i], R[i] =   0.9967 R[i-1] + eps[i],
eps ~ N(0,  0.3871^2)
##          (0.0230)        (1.8559)                (0.0059) ##
## R[687] = -0.1228 (t = -0.049) ##
## WARNING: X does not seem to be integrated. X and Y do not appear to

be cointegrated. summary(EGCM)

## Y[i] =   0.2329 X[i] -  14.7253 + R[i], R[i] =   0.9967 R[i-1] + eps[i],
eps ~ N(0,  0.3871^2)
##          (0.0230)        (1.8559)                (0.0059) ##
## R[687] = -0.1228 (t = -0.049) ##
## WARNING: X does not seem to be integrated. X and Y do not appear to be
cointegrated.
##
## Unit Root Tests of Residuals
##                                                    Statistic    p-value
##   Augmented Dickey Fuller (ADF)                       -3.355    0.04426
##   Phillips-Perron (PP)                               -15.152    0.15933
##   Pantula, Gonzales-Farias and Fuller (PGFF)           0.988    0.42319
##   Elliott, Rothenberg and Stock DF-GLS (ERSD)         -1.918    0.16510
##   Johansen's Trace Test (JOT)                        -39.357    0.00010
##   Schmidt and Phillips Rho (SPR)                     -30.945    0.02913
##
## Variances
##   SD(diff(X))          =   0.734868
##   SD(diff(Y))          =   0.383570
##   SD(diff(residuals))  =   0.387718
##   SD(residuals)        =   2.521711
##   SD(innovations)      =   0.387119
##
## Half life       = 211.728765
## R[last]         =  -0.122817 (t=-0.05)
```

Graphic 3 (EGCM Graphs:)

```
plot(EGCM)

## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use
"none" instead as
## of ggplot2 3.3.4.
## ℹ The deprecated feature was likely used in the egcm package.
##   Please report the issue to the authors.
## This warning is displayed once every 8 hours.
```

```
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was ## generated.
```



```r
#Engle-Granger Methodology (using package "aTSA")
coint.test(Inflation, TCU,           d = 0, nlag = NULL, output = TRUE)
# in levels
```

```
## Response: Inflation
## Input: TCU
## Number of inputs: 1
## Model: y ~ X + 1
## -------------------------------
## Engle-Granger Cointegration Test
## alternative: cointegrated
##
## Type 1: no trend
##     lag      EG p.value  ##
6.000  -3.096   0.034
## -----
##  Type 2: linear trend
##     lag      EG p.value  ##
6.000  -0.544   0.100
## -----
##  Type 3: quadratic trend
##      lag       EG  p.value  ##
6.00000 -0.00024  0.10000
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

Output shows “Not co integrated,” signaling no stable long-term relationship between CPI Inflation and TCU, complicating policy forecasting. Johansen Test refines Engle-Granger for multi-series analysis.

```r
VARselect(data.set, lag.max=5, type="none",  season = NULL, exogen =
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      2      2      2      2
```

```r
VARselect(data.set, lag.max=5, type="const", season = NULL, exogen =
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      2      2      2      2
```

```r
VARselect(data.set, lag.max=5, type="trend", season = NULL, exogen =
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      2      2      2      2
```

```r
VARselect(data.set, lag.max=5, type="both",  season = NULL, exogen = 
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n) 
##      2      2      2      2
```

```r
VARselect(data.set, lag.max=10, type="none",  season = NULL, exogen = 
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n) 
##      2      2      2      2
```

```r
VARselect(data.set, lag.max=10, type="const", season = NULL, exogen = 
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n) 
##      2      2      2      2
```

```r
VARselect(data.set, lag.max=10, type="trend", season = NULL, exogen = 
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n) 
##      2      2      2      2
```

```r
VARselect(data.set, lag.max=10, type="both",  season = NULL, exogen = 
NULL)$selection
```

```
## AIC(n)  HQ(n)  SC(n) FPE(n) 
##      2      2      2      2
```

The output shows that the Hannan-Quinn (HQ) criterion, due to the monthly data structure, identifies 2 as the optimal lag count. This lag is then applied to configure the dataset for the Autoregressive Distributed Lag (ARDL) model.

```r
optimal.lags = 2 # Determined by the Hannan-Quinn Information Criterion

# Conducting Johansen co-integration tests using the eigenvalue method
johansen.const = ca.jo(data.set, type="eigen", ecdet="const", K =
optimal.lags, spec="longrun") summary(johansen.const) # Summarizing the
results of the eigenvalue test

##
## ######################
## # Johansen-Procedure #  ##
######################
##
## Test type: maximal eigenvalue statistic (lambda max) , without linear
trend and constant in cointegration
##
## Eigenvalues (lambda):
## [1] 3.336875e-02 2.324225e-02 1.387779e-17 ##
## Values of teststatistic and critical values of test:
##
##            test 10pct  5pct  1pct
## r <= 1 | 16.11  7.52  9.24 12.97
## r = 0  | 23.25 13.75 15.67 20.20
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##               Inflation.l2      TCU.l2    constant
## Inflation.l2     1.000000  1.00000000    1.000000
## TCU.l2          -2.227963 -0.05442966    2.034051 ##
constant       173.829458  0.55765347 -278.154362 ##
## Weights W:
## (This is the loading matrix)
##
##              Inflation.l2      TCU.l2      constant
## Inflation.d -0.005095627 -0.01502379 -9.294576e-19
## TCU.d        0.009357644 -0.03210857  2.634836e-18

# Conducting Johansen co-integration tests using the trace statistic method
johansen.const = ca.jo(data.set, type="trace", ecdet="const", K =
optimal.lags, spec="longrun")
summary(johansen.const) # Summarizing the results of the trace statistic test
```

```
##
## ######################
## # Johansen-Procedure #  ##
######################
##
## Test type: trace statistic , without linear trend and constant in
cointegration
##
## Eigenvalues (lambda):
## [1] 3.336875e-02 2.324225e-02 1.387779e-17 ##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 1 | 16.11  7.52  9.24 12.97
## r = 0  | 39.36 17.85 19.96 24.60
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##                Inflation.l2      TCU.l2    constant
## Inflation.l2     1.000000  1.00000000    1.000000
## TCU.l2          -2.227963 -0.05442966    2.034051 ##
constant       173.829458  0.55765347 -278.154362 ##
## Weights W:
## (This is the loading matrix)
##
##               Inflation.l2      TCU.l2      constant
## Inflation.d -0.005095627 -0.01502379 -9.294576e-19
## TCU.d        0.009357644 -0.03210857  2.634836e-18
```

Johansen test results indicate long-term, stable co-integration among series, despite individual fluctuations.

ARDL Model:

The ARDL model assesses Inflation's response to TCU shifts, confirming precision with significant statistics.

```
# Initializing the random number generator for consistent Confidence
Interval simulation set.seed(123)
# HQ criterion determines 2 as the ideal lag number lags
= 2
# dynardl function takes a data frame, not a ts object
# Executing the dynardl function on the data.set data frame
ARDL <- dynardl(
                  formula = Inflation ~ TCU,
lags = list(TCU = 1, Inflation = 1),
```

```
                  diffs = "TCU",
                  lagdiffs = list(TCU = 1:lags, Inflation = 1:lags),
ec = TRUE, # Includes error correction                  constant =
TRUE, # Model includes a constant                  trend = FALSE, # No
trend component in the model                  simulate = TRUE, # Runs
simulations for shock response                  shockvar = "TCU", #
Variable for impulse response                  range = 50, # Forecast
horizon for impulse response                  sims = 1000, # Number of
simulations                  fullsims = TRUE, # Use full simulation
method                  data = data.set # Dataset used for the model
)
```

```
## [1] "Error correction (EC) specified; dependent variable to be run in
differences."
## [1] "TCU shocked by one standard deviation of TCU by default." ##
[1] "dynardl estimating ..."
##   |
|                                                                      |   0%
|
|==                                                                    |   3%
|
|===                                                                   |   4%
|
|====                                                                  |   6%
|
|=====                                                                 |   7%
|
|======                                                                |   9%
|
|=======                                                               |  10%
|
|========                                                              |  11%
|
|=========                                                             |  13%
|
|==========                                                            |  14%
|
|===========                                                           |  16%
|
|============                                                          |  17%
|
|=============                                                         |  19%
|
|==============                                                        |  20%
|
|===============                                                       |  21%
|
|================                                                      |  23%
```

```
|=================                                                     |  24%
|
|==================                                                    |  26%
|
|===================                                                   |  27%
|
|====================                                                  |  29%
|
|=====================                                                 |  30%
|
|======================                                                |  31%
|
|=======================                                               |  33%
|
|========================                                              |  34%
|
|=========================                                             |  36%
|
|==========================                                            |  37%
|
|===========================                                           |  39%
|
|============================                                          |  40%
|
|=============================                                         |  41%
|
|==============================                                        |  43%
|
|===============================                                       |  44%
|
|================================                                      |  46%
|
|=================================                                     |  47%
|
|==================================                                    |  49%
|
|===================================                                   |  50%
|
|====================================                                  |  51%
|
|=====================================                                 |  53%
|
|======================================                                |  54%
|
|=======================================                               |  56%
|
```

```
|========================================                              |  57%
|
|=========================================                             |  59%
|
|==========================================                            |  60%
|
|===========================================                           |  61%
|
|============================================                          |  63%
|
|=============================================                         |  64%
|
|==============================================                        |  66%
|
|===============================================                       |  67%
|
|================================================                      |  69%
|
|=================================================                     |  70%
|
|==================================================                    |  71%
|
|===================================================                   |  73%
|
|====================================================                  |  74%
|
|=====================================================                 |  76%
|
|======================================================                |  77%
|
|=======================================================               |  79%
|
|========================================================              |  80%
|
|=========================================================             |  81%
|
|==========================================================            |  83%
|
|===========================================================           |  84%
|
|============================================================          |  86%
|
|=============================================================         |  87%
|
|==============================================================        |  89%
|
```

```
|=================================================================       |  90%
|
|==================================================================      |  91%
|
|===================================================================     |  93%
|
|====================================================================    |  94%
|
|=====================================================================   |  96%
|
|======================================================================  |  97%
|
|======================================================================= |  99%
|
|========================================================================| 100%
```

```
# Displaying a summary of the ARDL model results summary(ARDL)
```

```
##
## Call:
## lm(formula = as.formula(paste(paste(dvnamelist), "~", paste(colnames(IVs),
##     collapse = "+"), collapse = " ")))
##
## Residuals:
##      Min       1Q   Median       3Q      Max  ##
-1.90686 -0.20306 -0.00037  0.19122  1.72154
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.033394   0.274245  -3.768 0.000179 ***
## l.1.Inflation  -0.017311   0.005340  -3.242 0.001245 **
## ld.1.Inflation  0.377344   0.038306   9.851  < 2e-16 ***
## ld.2.Inflation -0.048855   0.038308  -1.275 0.202632
## d.1.TCU         0.075732   0.018950   3.996 7.14e-05 ***
## l.1.TCU         0.013790   0.003506   3.933 9.25e-05 ***
## ld.1.TCU       -0.004882   0.019663  -0.248 0.803996    ##
ld.2.TCU        0.025566   0.019041   1.343 0.179819    ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Residual standard error: 0.3427 on 676 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.2108, Adjusted R-squared:  0.2026
## F-statistic: 25.79 on 7 and 676 DF,  p-value: < 2.2e-16
```

The ARDL model assesses how Inflation responds to TCU shifts, including error correction to track adjustments. Significant coefficients, strong t-values, high R-squared values, and minimal residuals confirm its precision.

```r
pssbounds(ARDL)
```

```
##
##  PESARAN, SHIN AND SMITH (2001) COINTEGRATION TEST  ##
##  Observations: 684
##  Number of Lagged Regressors (not including LDV) (k): 1
##  Case: 3 (Unrestricted intercept; no trend)  ##
```

```
##  ------------------------------------------------------
##  -                       F-test                       -
##  ------------------------------------------------------
##                  <------- I(0) ------------ I(1) ----->
##  10% critical value       4.04            4.78
##  5% critical value        4.94            5.73  ##
1% critical value       6.84            7.84  ##
##
##  F-statistic = 9.684
##  ------------------------------------------------------
##  -                       t-test                       -
##  ------------------------------------------------------
##                  <------- I(0) ------------ I(1) ----->
##  10% critical value       -2.57            -2.91
##  5% critical value        -2.86            -3.22 ##
1% critical value       -3.43            -3.82 ##
##
##  t statistic = -3.242
##  ------------------------------------------------------
##  F-statistic note: Asymptotic critical values used.  ##
t-statistic note: Asymptotic critical values used.

pssbounds(ARDL, restriction=TRUE)

##
##  PESARAN, SHIN AND SMITH (2001) COINTEGRATION TEST  ##
##  Observations: 684
##  Number of Lagged Regressors (not including LDV) (k): 1
##  Case: 2 (Intercept included in F-stat restriction; no trend)  ##
##  ------------------------------------------------------
##  -                       F-test                       -
##  ------------------------------------------------------
##                  <------- I(0) ------------ I(1) ----->
##  10% critical value       3.02            3.51
##  5% critical value        3.62            4.16  ##
1% critical value       4.94            5.58  ##
##
##  F-statistic = 6.471
##
##  ------------------------------------------------------  ##
F-statistic note: Asymptotic critical values used.
##  t-statistic note: Critical values do not currently exist for Case II.
```

The PSS cointegration test indicates long-term relationships between series like Inflation and TCU, with F-statistics highlighting persistent interactions and consistent t-statistics confirming this, aiding economic policy.

We then check the behaviour of the residuals which is demonstrated below:

```
dynardl.auto.correlated(ARDL) #Behaviour of Residuals

## 
##  ------------------------------------------------------ 
##  Breusch-Godfrey LM Test 
##  Test statistic: 2.437 
##  p-value: 0.119 
##  H_0: no autocorrelation up to AR 1  ## 
##  ------------------------------------------------------ 
##  Shapiro-Wilk Test for Normality 
##  Test statistic: 0.959 
##  p-value: 0 
##  H_0: residuals are distributed normal  ## 
##  ------------------------------------------------------ 
##  Log-likelihood: -233.988 
##  AIC: 485.976 
##  BIC: 526.728 
##  Note: AIC and BIC calculated with k = 8 on T = 684 observations. 
## 
##  ------------------------------------------------------ 
## Shapiro-Wilk test indicates we reject the null hypothesis of normality at
p < 0.01.
```

The Breusch-Godfrey LM Test, with a p-value of 0.119, shows no autocorrelation; ShapiroWilk indicates non-normal residuals. Log-likelihood, AIC, and BIC assess model quality.

```
# Assigning Total Capacity Utilization as 'x' and Inflation as
'y' x <- TCU y <- Inflation

# Conducting linear regression of Inflation on TCU without log transformation
regression_model <- lm(y ~ x)

# Retrieving the residuals for diagnostic tests
model_residuals <- resid(regression_model)  # Numeric residuals are crucial
for accuracy in diagnostics

# Diagnostic Tests for Regression Validity:

shapiro.test(model_residuals)  # Tests if residuals are normally distributed
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_residuals
## W = 0.84961, p-value < 2.2e-16 jarque.bera.test(model_residuals)  #
Another normality test for residuals

##
##  Jarque Bera Test
##
## data:  model_residuals
## X-squared = 430.62, df = 2, p-value < 2.2e-16 bds.test(model_residuals)
# Checks the i.i.d. assumption of residuals

##
##   BDS Test
##
## data:  model_residuals
##
## Embedding dimension =  2 3
##
## Epsilon for close points =  1.2609 2.5217 3.7826 5.0434  ##
## Standard Normal =
##       [ 1.2609 ] [ 2.5217 ] [ 3.7826 ] [ 5.0434 ]
## [ 2 ]    69.3503    50.1916    43.9310    42.0307 ##
[ 3 ]    99.4556    55.9236    44.5722    40.8231 ##
## p-value =
##       [ 1.2609 ] [ 2.5217 ] [ 3.7826 ] [ 5.0434 ]
## [ 2 ]          0          0          0          0 ##
[ 3 ]          0          0          0          0

bptest(regression_model)  # Examines the presence of heteroskedasticity in
residuals

##
##  studentized Breusch-Pagan test
##
## data:  regression_model
## BP = 1.5375, df = 1, p-value = 0.215

dwtest(regression_model)  # Looks for first-order autocorrelation in
residuals

##
##  Durbin-Watson test
##
## data:  regression_model
```

```
## DW = 0.023608, p-value < 2.2e-16
## alternative hypothesis: true autocorrelation is greater than 0
```

```
# Breusch-Godfrey tests to assess serial correlation using different
methodologies: bgtest(regression_model, order = 1, type = "Chisq")  # Chi-
square version for asymptotic cases
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1 ##
## data:  regression_model
## LM test = 669.79, df = 1, p-value < 2.2e-16
```

```
bgtest(regression_model, order = 1, type = "F")  # F-test version for smaller
sample sizes
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1 ##
## data:  regression_model
## LM test = 26623, df1 = 1, df2 = 684, p-value < 2.2e-16
```

```
bgtest(regression_model, order = 2, type = "Chisq")  # Chi-square test for
second-order serial correlation
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 2 ##
## data:  regression_model
## LM test = 671.35, df = 2, p-value < 2.2e-16
```

```
bgtest(regression_model, order = 2, type = "F")  # F-test version for
secondorder correlation in smaller samples
```

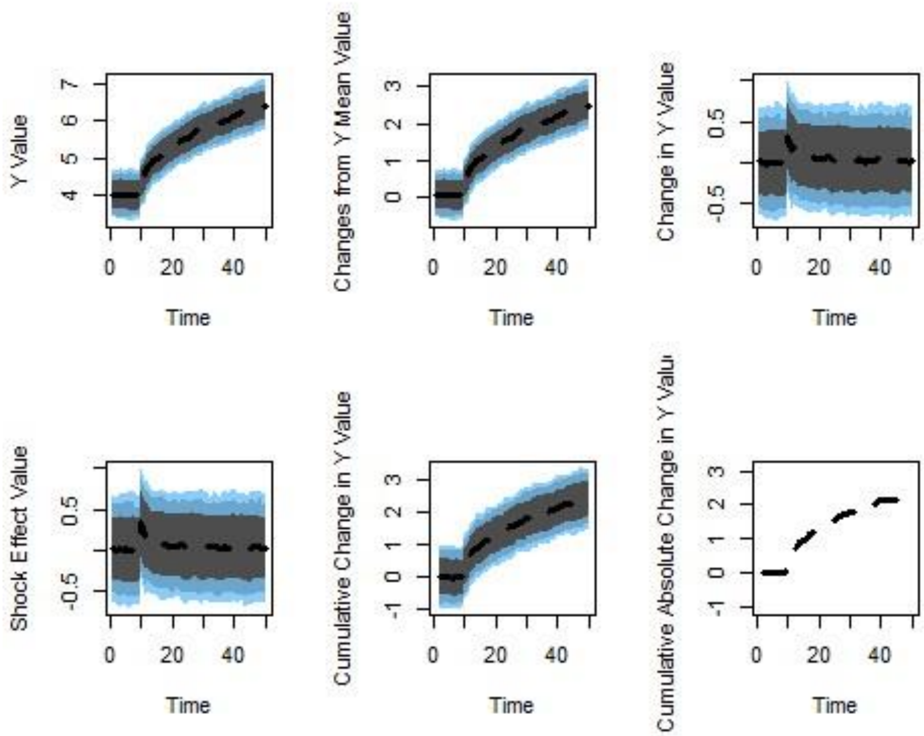
```
##
##  Breusch-Godfrey test for serial correlation of order up to 2 ##
## data:  regression_model
## LM test = 14652, df1 = 2, df2 = 683, p-value < 2.2e-16
```

Residuals fail standard tests, showing non-normality and serial correlation, likely affecting regression's reliability.

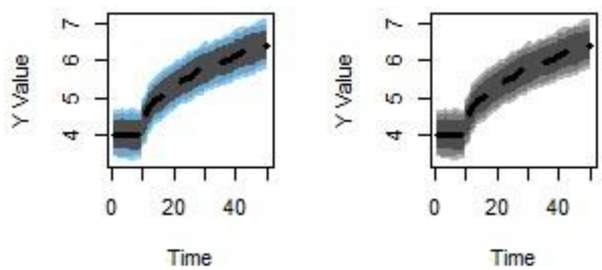
Graphic 3 (Impulse Response Plots)

```
#Impulse Response Plots dynardl.all.plots(ARDL)
# all plots together
```

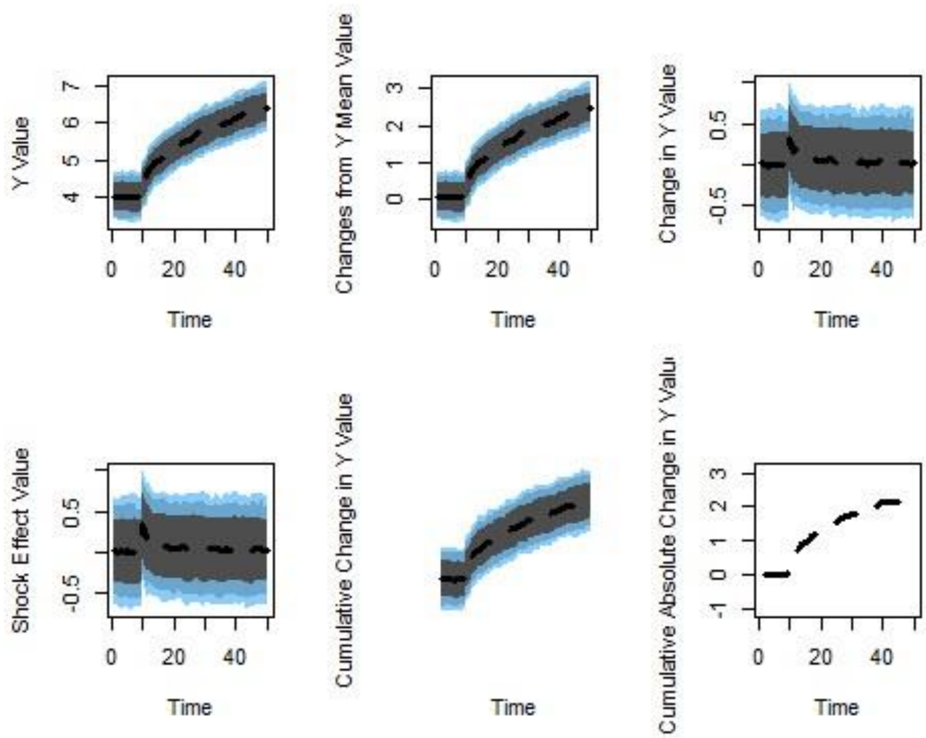
```
## Warning in dynardl.simulation.plot(x, response = "cumulative.abs.diffs", :
## Cumulative absolute effects assumed to be noise (by tolerance) at t = 40.
```




```
dynardl.simulation.plot(ARDL, type = "area", response = "levels")
# in colors dynardl.simulation.plot(ARDL, type = "area", response =
"levels", bw = TRUE) # in black and white




par(mfrow = c(2, 3))
```




```
   dynardl.simulation.plot(ARDL, type = "area", response = "levels")
dynardl.simulation.plot(ARDL, type = "area", response =
"levels.from.mean")
   dynardl.simulation.plot(ARDL, type = "area", response = "diffs")
dynardl.simulation.plot(ARDL, type = "area", response =
"shock.effect.decay")
   dynardl.simulation.plot(ARDL, type = "area", response =
"cumulative.diffs", axes = F)
   dynardl.simulation.plot(ARDL, type = "area", response =
"cumulative.abs.diffs")

## Warning in dynardl.simulation.plot(ARDL, type = "area", response =
## "cumulative.abs.diffs"): Cumulative absolute effects assumed to be noise (by
## tolerance) at t = 40.
```



Graphic 3 The impulse-response plots in the ARDL framework show how a shock in Total Capacity Utilization (TCU) affects Inflation, highlighting immediate impacts and long-term adjustments crucial for economic forecasting.

VAR model:

A VAR model examines how variables like Inflation and TCU interdependently influence each other over time.

```
optimal.lags = 2 # As shown from HQ criterion #Reduced
form VAR setting p = optimal lags
var.model.const <- VAR(data.set, p=optimal.lags, type="const",
exogen=NULL)#Constant but no trend    summary(var.model.const)
#Results

##
## VAR Estimation Results:
## =========================
## Endogenous variables: Inflation, TCU
## Deterministic variables: const
## Sample size: 685
## Log Likelihood: -955.616
## Roots of the characteristic
polynomial: ## 0.9734 0.9734 0.411 0.2376
## Call:
## VAR(y = data.set, p = optimal.lags, type = "const", exogen = NULL) ##
```

```
##
## Estimation results for equation Inflation:
## ==========================================
## Inflation = Inflation.l1 + TCU.l1 + Inflation.l2 + TCU.l2 + const
##
##              Estimate Std. Error t value Pr(>|t|)
## Inflation.l1  1.35913    0.03606  37.695  < 2e-16 *** ##
TCU.l1        0.03235    0.01865   1.735  0.08327 .
## Inflation.l2 -0.37925    0.03584 -10.583  < 2e-16 ***
## TCU.l2        -0.02018     0.01863   -1.083   0.27901
## const        -0.89415    0.27137  -3.295  0.00104 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.3466 on 680 degrees of freedom
## Multiple R-Squared: 0.9837,  Adjusted R-squared: 0.9836
## F-statistic: 1.026e+04 on 4 and 680 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation TCU:
## ====================================
## TCU = Inflation.l1 + TCU.l1 + Inflation.l2 + TCU.l2 + const  ##
##              Estimate Std. Error t value Pr(>|t|)
## Inflation.l1  0.17764    0.07226   2.458  0.01421 *
## TCU.l1        1.23557    0.03738  33.056  < 2e-16 ***
## Inflation.l2 -0.20039    0.07182  -2.790  0.00542 **
## TCU.l2       -0.25467    0.03733  -6.822 1.98e-11 *** ##
const         1.60873    0.54389   2.958  0.00321 **  ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.6948 on 680 degrees of freedom
## Multiple R-Squared: 0.9723,  Adjusted R-squared: 0.9722  ##
F-statistic:  5977 on 4 and 680 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##           Inflation     TCU
## Inflation   0.12016 0.03683
## TCU         0.03683 0.48268
##
## Correlation matrix of residuals:
##           Inflation    TCU
## Inflation    1.0000 0.1529
```

```
## TCU           0.1529 1.0000
```

VAR model reveals significant long-term equilibrium between Inflation and TCU, supported by Johansen tests for VECM.

```
# Bivariate Vector Error Correction Model (VEC) configuration:
# Determining optimal lag length based on the Hannan-Quinn Information
Criterion
optimal.lags <- 2

# Implementing the Johansen cointegration test with a constant term in the
cointegrating equation
# and specifying the model for long-run relationships analysis johansen.model
<- ca.jo(data.set, type="eigen", ecdet="const", K = optimal.lags,
spec="longrun")

# Displaying a summary of the Johansen test results to understand the
longterm equilibria summary(johansen.model)

##
## ######################
## # Johansen-Procedure #  ##
######################
##
## Test type: maximal eigenvalue statistic (lambda max) , without linear
trend and constant in cointegration
##
## Eigenvalues (lambda):
## [1] 3.336875e-02 2.324225e-02 1.387779e-17 ##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 1 | 16.11  7.52  9.24 12.97
## r = 0  | 23.25 13.75 15.67 20.20
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##                Inflation.l2      TCU.l2    constant
## Inflation.l2     1.000000  1.00000000    1.000000
## TCU.l2          -2.227963 -0.05442966    2.034051 ##
constant       173.829458  0.55765347 -278.154362 ##
## Weights W:
## (This is the loading matrix)
##
##               Inflation.l2      TCU.l2      constant
## Inflation.d -0.005095627 -0.01502379 -9.294576e-19
## TCU.d        0.009357644 -0.03210857  2.634836e-18
```

The Vector Error Correction Model (VECM)

The Johansen test indicates potential long-term equilibrium between Inflation and TCU, using eigenvalues and eigenvectors to quantify cointegration strength and speed of adjustment to disequilibria.

Analysis of two variables requires setting one cointegrating vector.

```
# Setting up a bivariate Vector Error Correction Model (VECM) with the 'urca' package
# Determine one cointegrating relationship between the two series Inflation and TCU
VECM = cajorls(johansen.const, r = 1) # Select one cointegrating vector for the pair of series print(VECM) # Display the VECM results

## $rlm ##
## Call:
## lm(formula = substitute(form1), data = data.mat) ##
## Coefficients:
##                 Inflation.d  TCU.d
## ect1            -0.005096     0.009358
## Inflation.dl1    0.365372     0.190984
## TCU.dl1          0.039241     0.250302
##
##
## $beta
##                     ect1
## Inflation.l2   1.000000
## TCU.l2        -2.227963
## constant     173.829458
```

The VECM shows how Inflation and TCU dynamically adjust and stabilize after deviations, revealing long-term inverse relationships and immediate reactions to shifts.

```
linear.trend.test = lttest(johansen.const, r=1) # Test for Linear Trend and Ho = no linear trend

## LR-test for no linear trend
##
## H0: H*2(r<=1)
## H1: H2(r<=1)
##
## Test statistic is distributed as chi-square
## with 1 degress of freedom
##         test statistic p-value
## LR test           0.03    0.85
```

The LR test, checking for linear trends, shows insufficient evidence to suggest significant trends, as diagnostics evaluate residual serial correlation.

```r
VECM = vec2var(johansen.const, r = 1)#Transform Johanssen regression with constant into VEC model


# Diagnostic Checks:


    # Serial correlation test
    # Ho = residuals do not have serial correlation


serialtest <- serial.test(VECM, type = "PT.asymptotic") serialtest

##
##  Portmanteau Test (asymptotic)
##
## data:  Residuals of VAR object VECM
## Chi-squared = 268.47, df = 58, p-value < 2.2e-16

serialtest <- serial.test(VECM, type = "PT.adjusted") serialtest

##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object VECM
## Chi-squared = 272.83, df = 58, p-value < 2.2e-16

serialtest <- serial.test(VECM, type = "BG") serialtest

##
##  Breusch-Godfrey LM test
##
## data:  Residuals of VAR object VECM
## Chi-squared = 21.962, df = 20, p-value = 0.3426

serialtest <- serial.test(VECM, type = "ES") serialtest

##
##  Edgerton-Shukur F test
##
## data:  Residuals of VAR object VECM
## F statistic = 1.09, df1 = 20, df2 = 1338, p-value = 0.3531
```

Portmanteau tests show significant serial correlation; Breusch-Godfrey and EdgertonShukur F tests do not. Granger-Causality test requires stationarity.

```
# ---------- Toda-Yamamoto Version 2 (the best version)-------------

   Toda.Yamamoto = function(var) {

       # Add the extra lag to the VAR model:

     ty.df              =  eval(var$call$y);
ty.varnames       =  colnames(ty.df);     ty.lags
=  var$p + 1;
     ty.augmented_var  =  VAR(ty.df, ty.lags, type=var$type);

     ty.results = data.frame(predictor = character(0), causes = character(0),
chisq = numeric(0), p = numeric(0));

     for (current_variable in ty.varnames) {

        # Construct the restriction matrix to test if "current_variable"
causes any of the other variables.
        # We test if the lagged values of the "current variable" (ignoring
the extra lag) are jointly insignificant.

    ty.restrictions = as.matrix(Bcoef(ty.augmented_var))*0+1;
    ty.coefres       = head(grep(current_variable, colnames(ty.restrictions),
value=T), -1);
    ty.restrictions[which(rownames(ty.restrictions)   !=   current_variable),
ty.coefres] = 0;

        # Estimate the restricted VAR:

    ty.restricted_var = restrict(ty.augmented_var, 'manual',
resmat=ty.restrictions);

    for (k in 1:length(ty.varnames)) {

      if (ty.varnames[k] != current_variable) {

        my.wald    = waldtest(ty.augmented_var$varresult[[k]],
ty.restricted_var$varresult[[k]], test='Chisq');

        ty.results = rbind(ty.results, data.frame(

                                          predictor  =  current_variable,
causes  =  ty.varnames[k],                                        chisq
=  as.numeric(my.wald$Chisq[2]),
                                                    p  =
my.wald$`Pr(>Chisq)`[2])
                                                    );
```

```
        }}}
      return(ty.results);         }
```

The output provided shows the VAR model measured in levels which shows stationary data.

```
    # Ho: no Granger causality
    # H1: Granger-causality present
  # p-value lower than 5% indicates Granger-Causality
# Run the Toda-Yamamoto causality test:
   # Note that this part uses the VAR model you estimated
before.        var = var.model.const        Toda.Yamamoto(var)

##   predictor    causes    chisq          p
## 1 Inflation       TCU 7.934087 0.01892931
## 2       TCU Inflation 7.094694 0.02880094
```

Toda-Yamamoto shows mutual causality between Inflation and TCU; Cholesky Decomposition aids SVAR by improving linear equation solving.

```
# Arrange the dataset by specifying the order of variables for Cholesky
Decomposition
# First ordering: Inflation impacts TCU
ordered.data.set.inflation_first = data.set[, c("Inflation", "TCU")]

# Second ordering: TCU impacts Inflation
ordered.data.set.tcu_first = data.set[, c("TCU", "Inflation")]

# Conduct Johansen cointegration tests using both orderings for the SVAR
model
# with a constant for the long-term trend
johansen.inflation_first = ca.jo(ordered.data.set.inflation_first,
type="eigen", ecdet="const", K = optimal.lags, spec="longrun")
summary(johansen.inflation_first)

##
## ######################
## # Johansen-Procedure #  ##
######################
##
## Test type: maximal eigenvalue statistic (lambda max) , without linear
trend and constant in cointegration
##
## Eigenvalues (lambda):
## [1] 3.336875e-02 2.324225e-02 1.387779e-17 ##
## Values of teststatistic and critical values of test:
```

```
##
##           test 10pct  5pct  1pct
## r <= 1 | 16.11  7.52  9.24 12.97
## r = 0  | 23.25 13.75 15.67 20.20
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##               Inflation.l2      TCU.l2    constant
## Inflation.l2     1.000000  1.00000000    1.000000
## TCU.l2          -2.227963 -0.05442966    2.034051 ##
constant       173.829458  0.55765347 -278.154362 ##
## Weights W:
## (This is the loading matrix)
##
##             Inflation.l2      TCU.l2      constant
## Inflation.d -0.005095627 -0.01502379 -9.294576e-19 ##
TCU.d        0.009357644 -0.03210857  2.634836e-18
```

```
johansen.tcu_first = ca.jo(ordered.data.set.tcu_first, type="eigen",
ecdet="const", K = optimal.lags, spec="transitory")
summary(johansen.tcu_first)
```

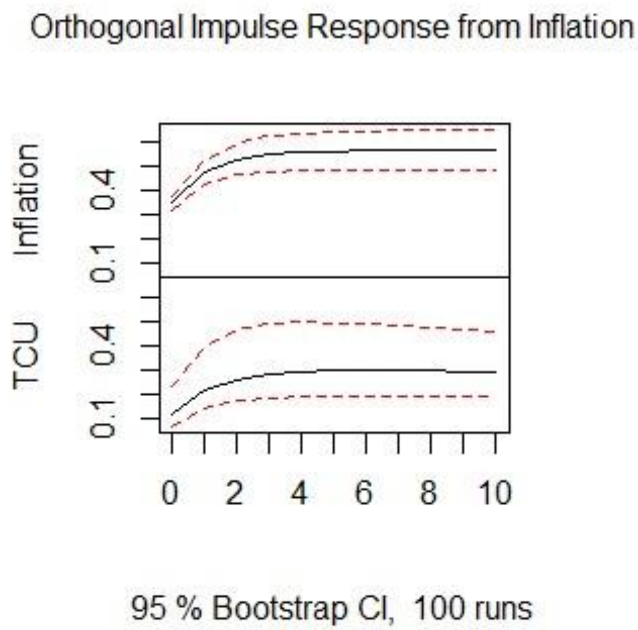
```
##
## ######################
## # Johansen-Procedure #  ##
######################
##
## Test type: maximal eigenvalue statistic (lambda max) , without linear
trend and constant in cointegration
##
## Eigenvalues (lambda):
## [1] 0.03336875 0.02324225 0.00000000 ##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 1 | 16.11  7.52  9.24 12.97
## r = 0  | 23.25 13.75 15.67 20.20
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##                    TCU.l1 Inflation.l1     constant
## TCU.l1          1.0000000      1.00000    1.0000000
## Inflation.l1   -0.4488405    -18.37234    0.4916297 ##
constant      -78.0216955    -10.24540 -136.7489408 ##
## Weights W:
```

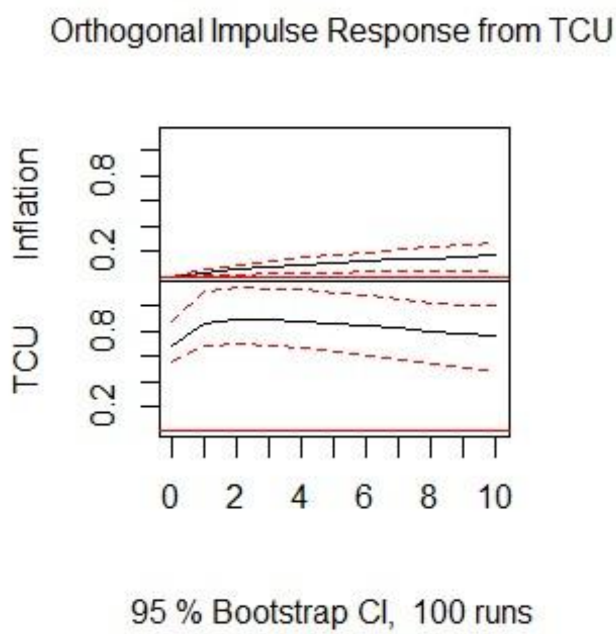
```
## (This is the loading matrix)
##
##                  TCU.l1 Inflation.l1      constant
## TCU.d       -0.02084849  0.001747658 -9.834251e-18
## Inflation.d  0.01135287  0.000817740  5.752766e-18
```

```r
# Convert the Johansen cointegration results into a VECM with one
cointegrating relationship
VECM.inflation_first = vec2var(johansen.inflation_first, r = 1)
VECM.tcu_first = vec2var(johansen.tcu_first, r = 1)
```

Graphic 4: Impulse Response Plots for VEC (Non-Cumulative)

```r
#Short-Run Non-Cumulative plot(irf(VECM))
```

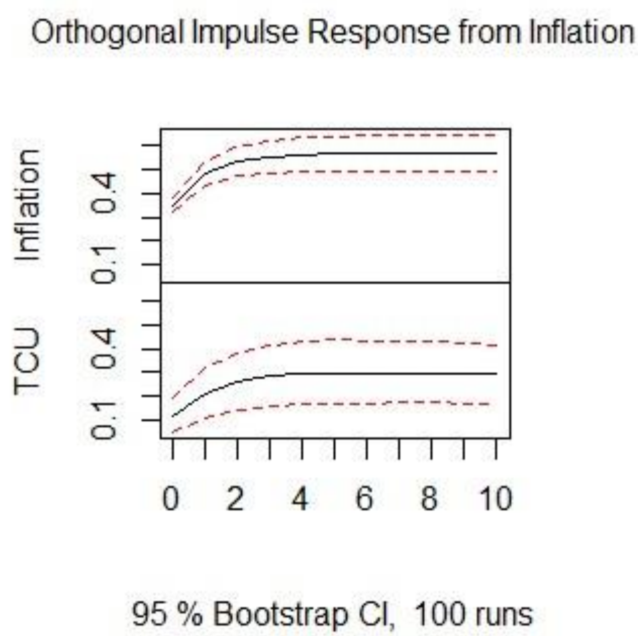




Graphic 5: Impulse Response Plots for VEC (Cumulative)

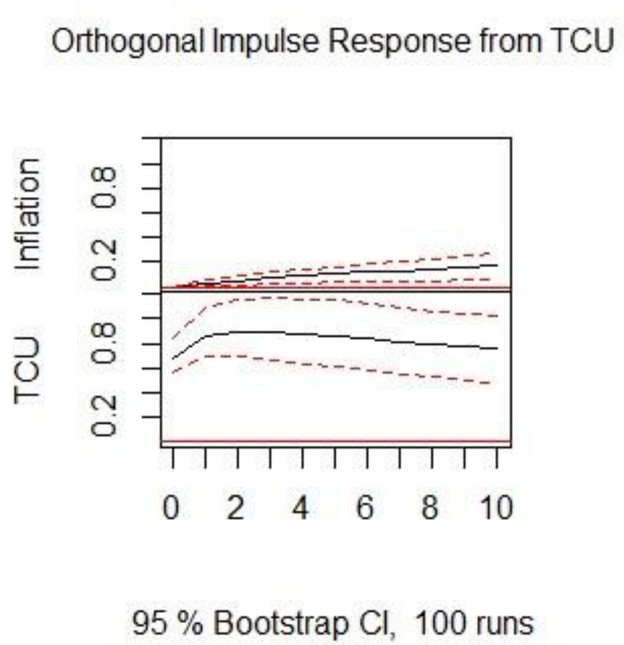
```
#Long run (cumulative):
plot(irf(VECM), cumulative=TRUE)
## Warning in plot.window(...): "cumulative" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "cumulative" is not a graphical
parameter

## Warning in title(...): "cumulative" is not a graphical parameter

## Warning in plot.window(...): "cumulative" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "cumulative" is not a graphical
parameter

## Warning in title(...): "cumulative" is not a graphical parameter

## Warning in mtext(main, 3, line = 2, outer = TRUE, adj = adj.mtext, padj
= ## padj.mtext, : "cumulative" is not a graphical parameter

## Warning in mtext(sub, 1, line = 4, outer = TRUE, adj = adj.mtext, padj =
## padj.mtext, : "cumulative" is not a graphical parameter
```




```
## Warning in plot.window(...): "cumulative" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "cumulative" is not a graphical
parameter

## Warning in title(...): "cumulative" is not a graphical parameter

## Warning in plot.window(...): "cumulative" is not a graphical parameter
## Warning in plot.xy(xy, type, ...): "cumulative" is not a graphical
parameter

## Warning in title(...): "cumulative" is not a graphical parameter
```

```
## Warning in mtext(main, 3, line = 2, outer = TRUE, adj = adj.mtext, padj
= ## padj.mtext, : "cumulative" is not a graphical parameter

## Warning in mtext(sub, 1, line = 4, outer = TRUE, adj = adj.mtext, padj =
## padj.mtext, : "cumulative" is not a graphical parameter
```



ARIMA model

ARIMA merges autoregressive and moving average methods, analyzing historical data and errors for accurate future forecasts.

```
   # Select a time series:
      # We need to use the time series in levels, even if they have a unit
root and thus are non-stationary
      # No need to first-difference the data at this point
z = Inflation # One of the Time Series
```

The code below will denote the best ARIMA model

```
    # "forecast::auto.arima()" means you are using function "auto.arima()"
from package "forecast"  arima.model = forecast::auto.arima(z,
```

```
                                      D = 1,
stationary = FALSE,
                                    ic = c("aicc", "aic", "bic"),
stepwise = FALSE,                         approximation = FALSE,
seasonal = TRUE,                            allowdrift = TRUE
)  arima.model

## Series: z
## ARIMA(2,0,0)(2,1,0)[12]
##
## Coefficients:
##          ar1      ar2     sar1     sar2
##       1.4620  -0.4823  -0.9837  -0.5195
## s.e.  0.0342   0.0341   0.0338   0.0333
##
## sigma^2 = 0.1533:  log likelihood = -331.44
## AIC=672.89   AICc=672.98   BIC=695.46
```

The displayed findings suggest the optimal ARIMA model for series Z is (2,0,0) with monthly seasonal adjustments at (2,1,0).

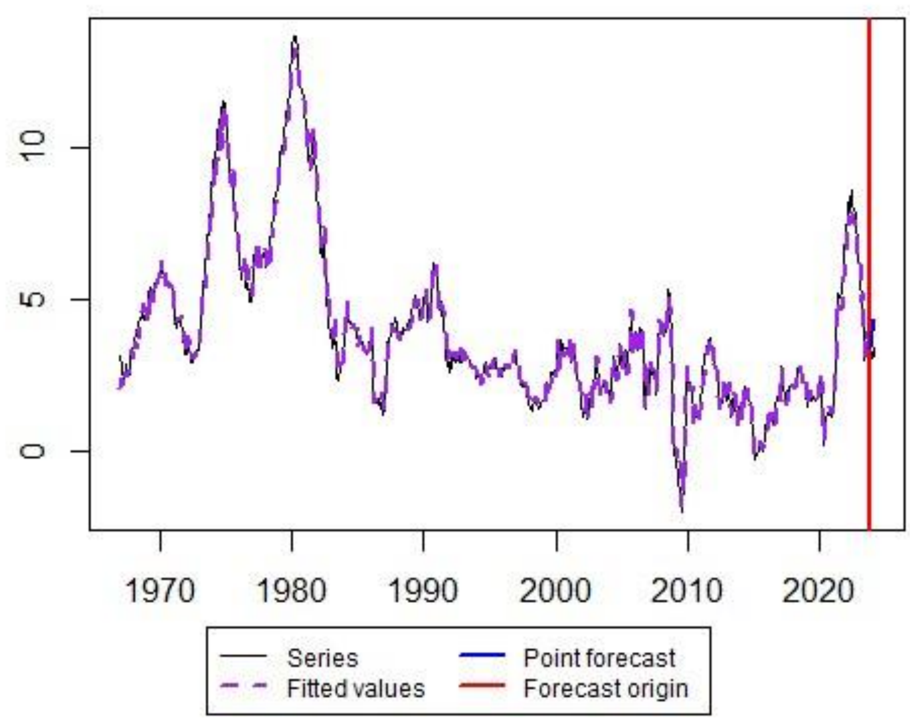
Graphic 6: In Sample Forecast

```
sarima.model = smooth::msarima(z, orders=list(ar = c(2,2), i = c(0,1), ma =
c(1,0)),
                               lags = c(1,12),
                               h = 5, # New, shorter in-sample forecast
period
                               holdout = TRUE)
summary(sarima.model)

## Warning: Observed Fisher Information is not positive semi-definite, which
means
## that the likelihood was not maximised properly. Consider reestimating the
## model, tuning the optimiser or using bootstrap via bootstrap=TRUE.

##
## Model estimated using msarima() function: SARIMA(2,0,1)[1](2,1,0)[12] ##
Response variable: data
## Distribution used in the estimation: Normal
## Loss function type: likelihood; Loss function value: 489.2858 ##
Coefficients:
##              Estimate Std. Error Lower 2.5% Upper 97.5%
## phi1[1]        0.1000     0.0014     0.0972      0.1028 *
## phi2[1]        0.7010     0.0016     0.6978      0.2800 *
## theta1[1]      0.9126     0.0789     0.7576      1.0675 *
## phi1[12]      -0.6152     0.0050     0.2500     -0.6053 *
## phi2[12]      -0.2125     0.0034    -0.2193     -0.2057 *
## ARIMAState1   -0.0368     2.8278    -5.5897      5.5148
## ARIMAState2   -0.0820     2.0898    -4.1857      4.0208
```

```
## ARIMAState3    0.1096     5.4230   -10.5394     10.7562
## ARIMAState4    0.1492     7.4530   -14.4861     14.7812
## ARIMAState5    0.1957     6.2450   -12.0675     12.4561
## ARIMAState6    0.1255    13.5177   -26.4191     26.6639
## ARIMAState7   -0.0173    17.3073   -34.0035     33.9610
## ARIMAState8   -0.1271    16.0528   -31.6499     31.3884
## ARIMAState9   -0.0738    20.3505   -40.0358     39.8789
## ARIMAState10  -0.1838    16.1977   -31.9910     31.6161
## ARIMAState11  -0.1438    18.6605   -36.7872     36.4912
## ARIMAState12  -0.0675     8.6480   -17.0494     16.9105
## ARIMAState13  -0.0237    10.2800   -20.2104     20.1583
## ARIMAState14   0.1750     5.8216   -11.2569     11.6043
## ARIMAState15  -0.0032     9.2431   -18.1538     18.1432
## ARIMAState16  -0.1400    10.7892   -21.3266     21.0416
## ARIMAState17  -0.1799    11.3596   -22.4866     22.1217
## ARIMAState18  -0.3419    22.5267   -44.5772     43.8833
## ARIMAState19  -0.2766    27.5274   -54.3319     53.7662
## ARIMAState20   0.1567    23.3689   -45.7325     46.0354
## ARIMAState21  -0.0280    28.8043   -56.5907     56.5217
## ARIMAState22   0.2054    21.6253   -42.2600     42.6610
## ARIMAState23   0.0912    24.4037   -47.8301     48.0014
## ARIMAState24  -0.0840     9.3539   -18.4523     18.2800
## ARIMAState25  -0.1871     7.6551   -15.2194     14.8416
## ARIMAState26   0.1481     4.5837    -8.8529      9.1469
## ARIMAState27  -0.6148     5.6777   -11.7640     10.5320
## ARIMAState28  -0.1054     7.2935   -14.4275     14.2135
## ARIMAState29  -0.8121     7.0200   -14.5971     12.9698
## ARIMAState30   0.0041     9.5717   -18.7918     18.7957
## ARIMAState31  -0.0372     8.8280   -17.3726     17.2943
## ARIMAState32  -0.6757    12.8494   -25.9080     24.5507
## ARIMAState33  -0.9462    11.4178   -23.3671     21.4696
## ARIMAState34  -0.8000    16.9897   -34.1624     32.5548
## ARIMAState35  -0.3177    13.8546   -27.5239     26.8821
## ARIMAState36   0.2442    23.2567   -45.4248     45.9027
## ARIMAState37   1.0045    17.6780   -33.7095     35.7106   ##
ARIMAState38   0.2394     8.0494   -15.5672     16.0422   ##
## Error standard deviation: 0.5126
## Sample size: 682
## Number of estimated parameters: 44
## Number of degrees of freedom: 638 ##
Information criteria:
##      AIC     AICc      BIC     BICc  ##
1066.572 1072.788 1265.673 1285.955
```

```
values = sarima.model
greybox::graphmaker(z,values$forecast,values$fitted,values$lower,values$upper
```

```
,level=0.95,legend=TRUE)
```



Graphic 6 SARIMA model graph shows actual data and fitted values, projecting future trends with confidence intervals.

Panel Data

Panel data tracks multiple subjects over time, enhancing analysis diversity and reducing aggregation bias.

part 2 optional

```
#Load the Packages
  library(lmtest)
library(texreg)              #

## Version:  1.39.3
## Date:     2023-11-09
## Author:   Philip Leifeld (University of Essex) ##
## Consider submitting praise using the praise or praise_interactive
functions.
## Please cite the JSS article in your publications -- see
citation("texreg").          library(tidyr)              # For tables and
summaries of outputs
```

```
##
## Attaching package: 'tidyr'

## The following object is masked from 'package:texreg':
##
##     extract          library(dplyr)             # For tables and
data manipulation

##
## ######################### Warning from 'xts' package
##########################
## #
#
## # The dplyr lag() function breaks how base R's lag() function is supposed
to  #
## # work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or
#
## # source() into this session won't work correctly.
#
## #
#
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can
add #
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop
#
## # dplyr from breaking base R's lag() function.
#
## #
#
## # Code in packages is not affected. It's protected by R's namespace
mechanism #
## # Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this warning.
#
## #
#
##
###########################################################################
##

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:xts':
##
##     first, last

## The following object is masked from 'package:car':
##
##     recode
```

```
## The following object is masked from 'package:MASS':
##
##     select

## The following object is masked from 'package:nlme':
##
##     collapse

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

          library(pdfetch)            # to fetch data directly from online
data bases
          library(foreign)            # Panel data plots and
visualizations          library(car)                # For plots
library(gplots)             # For plots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

          library(tseries)            # For time series and unit root tests
library(sjPlot)             # For consolidated regression tables

## #refugeeswelcome          library(huxtable)           # For

consolidated regression tables

##
## Attaching package: 'huxtable'

## The following object is masked from 'package:sjPlot':
##
##     font_size

## The following object is masked from 'package:dplyr':
##
##     add_rownames

          library(ivreg)              # For 2SLS and instrumental variables
(IV)          library(plm)                # Panel Data
Models

##
```

```
## Attaching package: 'plm'
```

```
## The following objects are masked from 'package:dplyr': ##
##     between, lag, lead

wdi = read.csv("wdi.csv", na.strings = "NA") # Load dataset wdi =
pdata.frame(wdi, index = c("Country", "Year"))    # transform the dataset
into a panel dataset 9
View(wdi)#View Entire Dataset in Alphabetical Order (Afghanistan-
Zimbabwe)
```

The dataset encompasses global data from the World Bank spanning from 1960 to 2015, covering all 193 United Nations member countries. It facilitates an in-depth analysis of maternal mortality rates worldwide and investigates an array of contributing factors, including GDP growth, access to healthcare, poverty levels, and other development indicators.

The code models infant mortality rates using health and economic factors within countries over time.

```
# Define variables for analysis within the panel data model
# InfantMortality: mortality rate, infant, per 1,000 live births
# OilRents: oil rents, constant 2010 US$
# PregnantWomenWithAnemia: total health expenditure as % of GDP
# NursesMidwives: nurses and midwives per 1,000 people
# SafeWaterAccess: percentage of population with access to safe water

# Create a panel data model using the 'within' estimator model <-
plm(InfantMortality ~ OilRents + PregnantWomenWithAnemia +
NursesMidwives + SafeWaterAccess,
data = wdi,                model =
"within")

# Display the summary of the model to interpret the results summary(model)

## Oneway (individual) effect Within Model
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess, data = wdi, model = "within") ##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Residuals:
##       Min.    1st Qu.     Median    3rd Qu.       Max.  ##
-15.64103  -0.61945   0.00000   0.60886  13.71773  ##
## Coefficients:
##                           Estimate Std. Error  t-value  Pr(>|t|)
## OilRents                  0.105916   0.067376   1.5720 0.1167855
## PregnantWomenWithAnemia  0.397542   0.106841   3.7209 0.0002288 ***
```

```
## NursesMidwives           0.034130   0.106682   0.3199 0.7492031     ##
SafeWaterAccess         -1.377345   0.084533 -16.2937 < 2.2e-16 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    10085
## Residual Sum of Squares: 3458.9
## R-Squared:      0.65702
## Adj. R-Squared: 0.49417
## F-statistic: 180.544 on 4 and 377 DF, p-value: < 2.22e-16
```

The model assesses the impact of various factors on infant mortality across different entities over time. It suggests that while oil rents show no significant effect, pregnant women with anemia have a notable positive impact, and increased safe water access significantly decreases infant mortality rates. The model is statistically significant with a strong fit as indicated by the F-statistic.

Random Effects:

Random effects models assume that the individual-specific effects are uncorrelated with the regressors across all time periods, treating these effects as part of the error term. This approach is useful for analyzing variations within groups across different entities when not all individual-specific variations are of interest.

```
RE_twoways = plm( InfantMortality ~ OilRents + PregnantWomenWithAnemia +
NursesMidwives + SafeWaterAccess,

                          data   = wdi,
model  = "random",
effect = "twoways",
random.method = "swar" )   summary(RE_twoways)

## Twoways effects Random Effect Model
##    (Swamy-Arora's transformation)
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess, data = wdi, effect =
"twoways",  ##     model = "random", random.method = "swar") ##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Effects:
##                    var std.dev share
## idiosyncratic   8.112   2.848 0.060
## individual    122.202  11.055 0.907
## time            4.348   2.085 0.032 ##
theta:
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
```

```
## id    0.7504989 0.8528650 0.8722312 0.8599907 0.8722312 0.9187932
## time  0.1931357 0.7780352 0.8234703 0.8044473 0.8317027 0.8823868 ## total
0.1929926 0.7612518 0.7927535 0.7734869 0.8067087 0.8681552 ##
## Residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.  ## -
35.278  -6.090  -2.920  -0.365   2.982  50.258  ##
## Coefficients:
##                          Estimate Std. Error  z-value Pr(>|z|)
## (Intercept)             99.772235   2.296819  43.4393  < 2e-16 ***
## OilRents                 0.025476   0.017074   1.4921  0.13568
## PregnantWomenWithAnemia  0.695198   0.025949  26.7908  < 2e-16 ***
## NursesMidwives            -0.076992   0.033264  -2.3146  0.02064 *   ##
SafeWaterAccess         -1.083380   0.018895 -57.3380  < 2e-16 *** ## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    413940
## Residual Sum of Squares: 77934
## R-Squared:      0.81539
## Adj. R-Squared: 0.81405
## Chisq: 8985.78 on 4 DF, p-value: < 2.22e-16
```

table

The Twoways effects Random Effect Model analyzes infant mortality using Swamy-Arora's transformation, addressing individual and time variations. Significant predictors include Pregnant Women with Anemia and Safe Water Access, both greatly impacting mortality rates. The model's high R-squared value (0.81539) indicates strong explanatory power, with a robust overall fit confirmed by a significant Chi-square statistic.

Fixed Effects:

Fixed effects model controls for unique characteristics influencing the outcome across individual or group entities.

```
#FE (Individual)
 FE_within_individual = plm( InfantMortality ~ OilRents +
PregnantWomenWithAnemia + NursesMidwives + SafeWaterAccess,

                      data   = wdi,
model  = "within",
effect = "individual")
summary(FE_within_individual)
```

```
## Oneway (individual) effect Within Model
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
```

```
##     NursesMidwives + SafeWaterAccess, data = wdi, effect = "individual",
##     model = "within")
##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Residuals:
##      Min.   1st Qu.    Median   3rd Qu.      Max.
## -15.64103  -0.61945   0.00000   0.60886  13.71773
##
## Coefficients:
##                           Estimate Std. Error  t-value  Pr(>|t|)
## OilRents                  0.105916   0.067376   1.5720 0.1167855
## PregnantWomenWithAnemia  0.397542   0.106841   3.7209 0.0002288 ***
## NursesMidwives            0.034130   0.106682   0.3199 0.7492031    ##
SafeWaterAccess         -1.377345   0.084533 -16.2937 < 2.2e-16 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    10085
## Residual Sum of Squares: 3458.9
## R-Squared:      0.65702
## Adj. R-Squared: 0.49417
## F-statistic: 180.544 on 4 and 377 DF, p-value: < 2.22e-16
```

The one-way fixed effects model, focusing on individual effects, analyzes factors influencing infant mortality. Significant findings show Pregnant Women With Anemia and Safe Water Access markedly affect mortality rates, as indicated by their t-values and p-values. The model's strong explanatory power is highlighted by an R-squared of 0.657.

```
#FE (Time)
 FE_within_time = plm( InfantMortality ~ OilRents + PregnantWomenWithAnemia +
NursesMidwives + SafeWaterAccess,

                      data = wdi,
model = "within",
effect = "time")

 summary(FE_within_time)
```

```
## Oneway (time) effect Within Model
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess, data = wdi, effect = "time",
##     model = "within")
##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Residuals:
##      Min.   1st Qu.    Median   3rd Qu.      Max.
```

```
## -33.58877  -5.65828  -0.53055   3.79266  48.36362
##
## Coefficients:
##                         Estimate Std. Error  t-value  Pr(>|t|)    ##
OilRents                -0.077336   0.039971  -1.9348  0.053541 .
## PregnantWomenWithAnemia  0.948416   0.058967  16.0838 < 2.2e-16 ***
## NursesMidwives           -0.465857   0.148505  -3.1370  0.001801 **
## SafeWaterAccess         -0.938393   0.041856 -22.4193 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    343270
## Residual Sum of Squares: 66412
## R-Squared:      0.80653
## Adj. R-Squared: 0.79969
## F-statistic: 559.672 on 4 and 537 DF, p-value: < 2.22e-16
```

Time-fixed effects analysis: The one-way time effect model for infant mortality highlights significant influences of Pregnant Women With Anemia and Safe Water Access, with substantial negative coefficients for these predictors. Oil Rents and Nurses Midwives also impact mortality, with Oil Rents showing marginal significance. The model demonstrates high explanatory power with an R-squared of 0.806.

Two-way fixed effects models in panel data analysis control for both entity-specific and time-specific unobserved variables. This approach allows for the removal of biases associated with omitted variables that vary across entities and over time, thereby providing more precise estimates of the effects of the observed variables on the outcome.

```r
#FE (Two-Ways)
FE_within_twoways = plm( InfantMortality ~ OilRents + PregnantWomenWithAnemia
+ NursesMidwives + SafeWaterAccess,

                          data = wdi,
model = "within",
effect = "twoways")
summary(FE_within_twoways)
```

```
## Twoways effects Within Model
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess, data = wdi, effect = "twoways",  ##
model = "within")
##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Residuals:
##        Min.     1st Qu.      Median     3rd Qu.        Max.  ##
-1.4658e+01 -6.0974e-01  2.0693e-04  6.3768e-01  1.2349e+01
##
## Coefficients:
##                          Estimate Std. Error  t-value Pr(>|t|)
## OilRents                 0.102451   0.067708   1.5131   0.1311
## PregnantWomenWithAnemia  0.185557   0.121807   1.5234   0.1285
## NursesMidwives           0.076740   0.102985   0.7452   0.4567    ##
SafeWaterAccess         -1.228185   0.095218 -12.8987   <2e-16 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    4787.5
## Residual Sum of Squares: 2936.6
## R-Squared:      0.38661
## Adj. R-Squared: 0.057882
## F-statistic: 57.04 on 4 and 362 DF, p-value: < 2.22e-16
```

The two-way fixed effects model analyzes the impact of various factors on infant mortality across different countries and time periods. Only Safe Water Access significantly decreases mortality (-1.228), with robust statistical backing (p < 2e-16). The model explains about 39% of the variance in infant mortality rates but has a lower adjusted R-squared, suggesting other unmodeled factors may also be important.

Pooled OLS

Pooled OLS treats panel data as a single dataset, merging all time periods and entities without accounting for individual or temporal differences. This simplification can lead to biased results because it overlooks potential variations and unique characteristics inherent in the data from different entities or time periods.

```
 pooled_model = plm( InfantMortality ~ OilRents + PregnantWomenWithAnemia +
NursesMidwives + SafeWaterAccess,

                     data        =   wdi,
model   = "pooling")


 summary(pooled_model)
```

```
## Pooling Model
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess, data = wdi, model = "pooling") ##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Residuals:
##      Min.   1st Qu.    Median   3rd Qu.      Max.
## -35.96696  -5.40263  -0.65018   3.51567  48.91508
##
## Coefficients:
##                          Estimate Std. Error  t-value  Pr(>|t|)
## (Intercept)             81.656494   4.957930  16.4699 < 2.2e-16 ***
## OilRents                -0.066376   0.040437  -1.6415  0.101270
## PregnantWomenWithAnemia  0.974250   0.058180  16.7454 < 2.2e-16 ***
## NursesMidwives          -0.444454   0.148052  -3.0020  0.002803 **
## SafeWaterAccess         -0.964827   0.042165 -22.8822 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    413940
## Residual Sum of Squares: 71677
## R-Squared:      0.82684
## Adj. R-Squared: 0.82559
## F-statistic: 658.959 on 4 and 552 DF, p-value: < 2.22e-16
```

The Pooled OLS model estimates the influence of various factors on infant mortality, showing Pregnant Women With Anemia and Safe Water Access significantly reduce mortality rates, demonstrated by robust t-values and extremely low p-values. The model's high R-squared (0.82684) indicates strong explanatory power, effectively capturing variability in infant mortality.

Hausman Test:

The Hausman test is used in panel data analysis to determine the more appropriate model between fixed effects and random effects. It tests if the unique errors (individual differences) are correlated with the independent variables. A significant test result suggests using a fixed effects model over a random effects model, ensuring that the analysis accounts for omitted variable biases that are correlated with the predictors.

```
# Testing correlation between unobserved effects and regressors in panel data
# Null hypothesis: No correlation; random effects model suitable
# Alternative hypothesis: Correlation present; fixed effects model necessary
hausman_test_individual = phtest(FE_within_individual, RE_twoways)
hausman_test_twoways = phtest(FE_within_twoways, RE_twoways)
```

Table

```
      # ---- Consolidate the results into a single table ---------------
----------------------------------------------------------------------
----------

      # Consolidate all model estimates into a single table, using
function "screenreg()" from package "texreg":

 screenreg(list(            pooled_model,
                        RE_twoways,
                        FE_within_individual,
                        FE_within_time,
                        FE_within_twoways ),

          custom.model.names = c("Pooled OLS",
```

```
                                    "Random Effects",
                                    "FE Within Individual",
                                    "FE Within Time",
                                    "FE Within Two-ways") )
```

```
##
##
=============================================================================
================================
##                           Pooled OLS  Random Effects  FE Within Individual
FE Within Time  FE Within Two-ways
## -----------------------------------------------------------------------------
-----------------------------------
## (Intercept)                81.66 ***   99.77 ***
##                            (4.96)      (2.30)
## OilRents                   -0.07        0.03            0.11
-0.08            0.10
##                            (0.04)      (0.02)          (0.07)
(0.04)          (0.07)
## PregnantWomenWithAnemia     0.97 ***    0.70 ***        0.40 ***
0.95 ***        0.19
##                            (0.06)      (0.03)          (0.11)
(0.06)          (0.12)
## NursesMidwives             -0.44 **    -0.08 *          0.03
-0.47 **         0.08
##                            (0.15)      (0.03)          (0.11)
(0.15)          (0.10)
## SafeWaterAccess            -0.96 ***   -1.08 ***       -1.38 ***
-0.94 ***       -1.23 ***
##                            (0.04)      (0.02)          (0.08)
(0.04)          (0.10)
## -----------------------------------------------------------------------------
----------------------------------
## R^2                         0.83        0.82            0.66
0.81            0.39
## Adj. R^2                    0.83        0.81            0.49
0.80            0.06
## Num. obs.                 557         557             557
557             557
## s_idios                                 2.85
## s_id                                   11.05
## s_time                                  2.09
##
=============================================================================
=================================
## *** p < 0.001; ** p < 0.01; * p < 0.05
```

This table compares pooled OLS, random effects, and two-way fixed effects models for Infant Mortality. The results highlight that Safe Water Access consistently decreases Infant

Mortality across models. Pregnant Women with Anemia is significant in all but the pooled model. The fixed effects models control for unobserved heterogeneity better than pooled OLS.

Diagnostics Checks

```
# Diagnostic tests ----------------------------------------------------------
--------------------------------------------------------------


       # Select a model first:

        model.selected = pooled_model
model.selected = RE_twoways
model.selected = FE_within_individual
model.selected = FE_within_time
model.selected = FE_within_twoways

       summary( model.selected )

## Twoways effects Within Model
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess, data = wdi, effect = "twoways",  ##
model = "within")
##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Residuals:
##        Min.     1st Qu.      Median     3rd Qu.        Max.  ##
-1.4658e+01 -6.0974e-01  2.0693e-04  6.3768e-01  1.2349e+01  ##
## Coefficients:
##                          Estimate Std. Error  t-value Pr(>|t|)
## OilRents                 0.102451   0.067708   1.5131   0.1311
## PregnantWomenWithAnemia  0.185557   0.121807   1.5234   0.1285
## NursesMidwives           0.076740   0.102985   0.7452   0.4567    ##
SafeWaterAccess         -1.228185   0.095218 -12.8987   <2e-16 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    4787.5
## Residual Sum of Squares: 2936.6
## R-Squared:      0.38661
## Adj. R-Squared: 0.057882
## F-statistic: 57.04 on 4 and 362 DF, p-value: < 2.22e-16

        pFtest( model.selected   , pooled_model       )               #
poolability tests

##
##  F test for twoways effects
```

```
##
## data:  InfantMortality ~ OilRents + PregnantWomenWithAnemia + NursesMidwives
+  ...
## F = 44.598, df1 = 190, df2 = 362, p-value < 2.2e-16
## alternative hypothesis: significant effects
```

```
       plmtest( model.selected    , effect="individual"  )                #
poolability tests
```

```
##
##  Lagrange Multiplier Test - (Honda) ##
## data:  InfantMortality ~ OilRents + PregnantWomenWithAnemia + NursesMidwives
+  ...
## normal = 18.907, p-value < 2.2e-16
## alternative hypothesis: significant effects
```

```
        phtest( model.selected   , RE_twoways            )              #
Hausman test
```

```
##
##  Hausman Test
##
## data:  InfantMortality ~ OilRents + PregnantWomenWithAnemia + NursesMidwives
+  ...
## chisq = 20.621, df = 4, p-value = 0.0003764 ##
alternative hypothesis: one model is inconsistent
```

```
       pbgtest( model.selected )                                      #
Breusch-Godfrey test for serial corelation
```

```
##
##  Breusch-Godfrey/Wooldridge test for serial correlation in panel models ##
## data:  InfantMortality ~ OilRents + PregnantWomenWithAnemia + NursesMidwives
+  ...
## chisq = 0.0040953, df = 1, p-value = 0.949
## alternative hypothesis: serial correlation in idiosyncratic errors
```

```
       pcdtest( model.selected )                                      #
Cross-sectional dependence (XSD) test
```

```
## Warning in pcdres(tres = tres, n = n, w = w, form =
paste(deparse(x$formula)),
## : Some pairs of individuals (130 percent) do not have any or just one time
## period in common and have been omitted from calculation
```

```
##
##  Pesaran CD test for cross-sectional dependence in panels ##
## data:  InfantMortality ~ OilRents + PregnantWomenWithAnemia +
NursesMidwives +     SafeWaterAccess
```

```
## z = 11.519, p-value < 2.2e-16
## alternative hypothesis: cross-sectional dependence
```

The two-way fixed effects model investigates the determinants of infant mortality, factoring in both individual and time variations. Only Safe Water Access significantly lowers infant mortality, with a robust negative effect. Tests indicate significant entity and time effects, no serial correlation, but there is cross-sectional dependence in the data.

Finding Instrumental Variables

```
     # ---- Baseline non-IV model:

      # Fixed effects (FE) model with two-way effects (individual and time
effects), with no instruments:
      # This is going to be our baseline model



  FE_within_twoways = plm( InfantMortality ~ OilRents +
PregnantWomenWithAnemia + NursesMidwives + SafeWaterAccess,

                          data = wdi,
model = "within",
effect = "twoways")


  summary(FE_within_twoways)

## Twoways effects Within Model
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess, data = wdi, effect = "twoways",
##     model = "within")
##
## Unbalanced Panel: n = 176, T = 1-10, N = 557
##
## Residuals:
##        Min.     1st Qu.      Median     3rd Qu.        Max.  ##
-1.4658e+01 -6.0974e-01  2.0693e-04  6.3768e-01  1.2349e+01  ##
## Coefficients:
##                           Estimate Std. Error  t-value Pr(>|t|)
## OilRents                  0.102451   0.067708   1.5131   0.1311
## PregnantWomenWithAnemia  0.185557   0.121807   1.5234   0.1285
## NursesMidwives            0.076740   0.102985   0.7452   0.4567    ##
SafeWaterAccess         -1.228185   0.095218 -12.8987   <2e-16 *** ## --
-
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    4787.5
## Residual Sum of Squares: 2936.6
## R-Squared:      0.38661
## Adj. R-Squared: 0.057882
## F-statistic: 57.04 on 4 and 362 DF, p-value: < 2.22e-16
```

In this two-way fixed effects model, after accounting for individual and time-specific effects, only Safe Water Access significantly predicts a decrease in infant mortality. Despite the model's reasonable fit, most variables lack statistical significance, implying limited explanatory power for the variations in infant mortality across the panel data.

```
      # Compute HAC and XSD robust standard errors:

  FE_within_twoways_HAC_Arellano = coeftest( FE_within_twoways,  vcov =
vcovHC(  FE_within_twoways,  method = "arellano" , type = "HC3") , save=TRUE
)  # HAC vcov from Arellano
  FE_within_twoways_HAC_DK       = coeftest( FE_within_twoways,  vcov =
vcovSCC( FE_within_twoways, cluster = "group"    , type = "HC3") , save=TRUE
)  # HAC vcov from Driscoll-Kraay, also robust to XSD
```

2SLS model

```
#Two-stage least squares (2SLS) with external IV
 FE_within_twoways_IV_TSLS = plm(InfantMortality ~ OilRents +
PregnantWomenWithAnemia + NursesMidwives + SafeWaterAccess #Regular model
                                | . - OilRents + GDPPerCapita +
IncomePerCapita + GDPGrowth,  # Instruments for
OilRents                                 data = wdi,
model = "within",
effect = "twoways",
                                inst.method = "bvk")     # Using 'bvk' method
for instrumental variables

summary(FE_within_twoways_IV_TSLS)

## Twoways effects Within Model
## Instrumental variable estimation
##
## Call:
## plm(formula = InfantMortality ~ OilRents + PregnantWomenWithAnemia +
##     NursesMidwives + SafeWaterAccess | . - OilRents + GDPPerCapita +
##     IncomePerCapita + GDPGrowth, data = wdi, effect = "twoways",
##     model = "within", inst.method = "bvk") ##
## Unbalanced Panel: n = 143, T = 1-10, N = 426
##
## Residuals:
##      Min.   1st Qu.    Median   3rd Qu.      Max.
## -14.17811  -0.58532   0.00000   0.70590  13.33785
##
## Coefficients:
```

```
##                             Estimate Std. Error z-value Pr(>|z|)
## OilRents                    -0.381173   0.305717 -1.2468  0.21246    ##
PregnantWomenWithAnemia  0.300822   0.170109  1.7684  0.07699 .
## NursesMidwives           0.078315   0.119056  0.6578  0.51067    ##
SafeWaterAccess         -1.191434   0.129629 -9.1911  < 2e-16 *** ##
---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 ##
## Total Sum of Squares:    3660.3
## Residual Sum of Squares: 2513.4
## R-Squared:      0.3184
## Adj. R-Squared: -0.097279
## Chisq: 143.903 on 4 DF, p-value: < 2.22e-16

          # Compute HAC and XSD robust standard errors:

      FE_within_twoways_IV_TSLS_HAC_Arellano = coeftest(
FE_within_twoways_IV_TSLS,  vcov = vcovHC(  FE_within_twoways_IV_TSLS,  method
= "arellano" , type = "HC3") , save=TRUE )  # HAC vcov from Arellano
      FE_within_twoways_IV_TSLS_HAC_DK       = coeftest(
FE_within_twoways_IV_TSLS,  vcov = vcovSCC( FE_within_twoways_IV_TSLS,
cluster = "group"    , type = "HC3") , save=TRUE )  # HAC vcov from Driscoll-
Kraay, also robust to XSD
```

The model uses instrumental variables to account for potential endogeneity in predicting infant mortality. While Safe Water Access significantly decreases infant mortality, other variables, including Oil Rents and NursesMidwives, are not statistically significant. Despite its complexity, the model's negative adjusted R-squared indicates potential overfitting or model misspecification.

GMM estimation

```
    GMM = pgmm(
InfantMortality ~
        lag(OilRents, 0:1) +      # Reduced number of lags for regressors
lag(PregnantWomenWithAnemia, 0:1) +        lag(NursesMidwives, 0:1)
    |
        lag(IncomePerCapita,  0:1)  +        #  Reduced  number  of  lags  for
instruments
        lag(GDPPerCapita, 0:1) +
lag(OilRents, 0:1) +
lag(GDPGrowth, 0:1),      data =
wdi,     model = "twosteps",
effect = "individual",
    transformation = "d"
)

## Warning in pgmm(InfantMortality ~ lag(OilRents, 0:1) +
## lag(PregnantWomenWithAnemia, : the first-step matrix is singular, a
```

```
general
## inverse is used

## Warning in pgmm(InfantMortality ~ lag(OilRents, 0:1) +
## lag(PregnantWomenWithAnemia, : the second-step matrix is singular, a
general
## inverse is used summary(GMM,

robust = TRUE)

## Warning in vcovHC.pgmm(object): a general inverse is used

## Oneway (individual) effect Two-steps model Difference GMM  ##
## Call:
## pgmm(formula = InfantMortality ~ lag(OilRents, 0:1) +
lag(PregnantWomenWithAnemia,
##     0:1) + lag(NursesMidwives, 0:1) | lag(IncomePerCapita, 0:1) +
##     lag(GDPPerCapita, 0:1) + lag(OilRents, 0:1) + lag(GDPGrowth,
##     0:1), data = wdi, effect = "individual", model = "twosteps",
##     transformation = "d")
##
## Balanced Panel: n = 192, T = 56, N = 10752 ##
## Number of Observations Used: 25 ##
Residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -2.52959  0.00000  0.00000 -0.00104  0.00000  1.30283
##
## Coefficients:
##                                    Estimate Std. Error z-value Pr(>|z|)
## lag(OilRents, 0:1)0               -0.031450   0.048163 -0.6530   0.5138
## lag(OilRents, 0:1)1               -0.044190   0.052398 -0.8434   0.3990
## lag(PregnantWomenWithAnemia, 0:1)0 -2.098890   1.885391 -1.1132   0.2656
## lag(PregnantWomenWithAnemia, 0:1)1  3.913920   2.823433  1.3862   0.1657
## lag(NursesMidwives, 0:1)0           0.050423   0.075364  0.6691   0.5035
## lag(NursesMidwives, 0:1)1           0.016083   0.018287  0.8795   0.3791
##
## Sargan test: chisq(430) = 8.809637 (p-value = 1)
## Autocorrelation test (1): normal = 0.9951402 (p-value = 0.31967)
## Autocorrelation test (2): normal = 0.996532 (p-value = 0.31899)
## Wald test for coefficients: chisq(6) = 329.9326 (p-value = < 2.22e-16)
```

The Difference GMM model attempts to estimate the effects of key predictors on infant mortality. However, the coefficients for lagged values of the predictors are not statistically significant. The Sargan test indicates valid instruments, but the presence of autocorrelation suggests the model’s dynamics may be misspecified, questioning the estimates’ reliability.

Conclusion

The study's thorough analysis revealed significant insights into the dynamic interplay between the Capacity Utilization Rate and CPI Inflation. While certain models indicated potential long-term equilibrium relationships, causality tests suggested a bidirectional influence between the variables. Discrepancies across models, flagged by diagnostic tests, highlighted the complexity of economic behaviors. The use of ARDL, VAR, and VEC models provided a nuanced view of the immediate and prolonged effects, contributing to a more informed economic forecasting and policy-making framework. Overall, the investigation underscored the intricate nexus of economic efficiency and inflationary trends, pivotal for understanding economic stability and growth.