Lab 5

Abdulla Almansoori, Alexander Firth, Nathan Seto

2025-04-10

#========== # A. Set up #==========

1.Set up working directory and clean global environment.
2.Load required libraries (to read excel files, to manipulate dataframes, to plot time series using ts.plot)

#========== # B. Define model variables #==========

```
# Input the sheet 'Monthly' of the file 'FinalData.xlsx
FinalData <- read_excel("~/ECON436_SalesTax/Data/FinalData.xlsx", sheet = 1)
FinalData$Lead_STF_Real[FinalData$year == 2024 & FinalData$Year != 12] <- FinalData$Lead_STF_Real[Finall
    2024 & FinalData$Year != 12] * (3.85/4.35)
FinalData$Lead_STF_Real[FinalData$year == 2025 & FinalData$Year == 1] <- FinalData$Lead_STF_Real[FinalDa
    2025 & FinalData$Year == 1] * (3.85/4.35)
FinalData$SG[1:12] <- FinalData$SG[1:12] + 5
FinalData$G[1:12] <- FinalData$G[1:12] + 5
FinalData$EDUHS[1:12] <- FinalData$EDUHS[1:12] - 5
FinalData <- FinalData[-nrow(FinalData), ]

Manufacturing <- read_excel("~/ECON436_SalesTax/Data/Manufacturing.xlsx")
Food_accomidation <- read_excel("~/ECON436_SalesTax/Data/Food and accomidation.xlsx")
retailemployees <- read_excel("~/ECON436_SalesTax/Data/retail employees.xlsx")
Transportation <- read_excel("~/ECON436_SalesTax/Data/Transportation.xlsx")

TrafficVolume <- read_excel("~/ECON436_SalesTax/Data/TrafficVolume.xlsx")
# Data from
# https://dtdapps.coloradodot.info/otis/TrafficData#ui/0/0/1/station/000127/criteria/27425//false/true/
TrafficVolume_clean <- TrafficVolume[TrafficVolume$Year >= 2012 & TrafficVolume$Year <=
    2024, ]
TrafficVolume_clean <- TrafficVolume_clean[, -c(1, 15)] |>
    arrange(Year)

long_Trans <- Transportation |>
    pivot_longer(cols = -Year, names_to = "Month", values_to = "transportation_employment") |>
    mutate(Month = match(Month, month.abb), .before = Month)

long_Retail <- retailemployees |>
    pivot_longer(cols = -Year, names_to = "Month", values_to = "RT_employment") |>
    mutate(Month = match(Month, month.abb), .before = Month)

long_Food <- Food_accomidation |>
    pivot_longer(cols = -Year, names_to = "Month", values_to = "AFS_employment") |>
    mutate(Month = match(Month, month.abb), .before = Month)
```

```
long_Manf <- Manufacturing |>
    pivot_longer(cols = -Year, names_to = "Month", values_to = "MAN_employment") |>
    mutate(Month = match(Month, month.abb), .before = Month)

long_Traffic <- TrafficVolume_clean |>
    pivot_longer(cols = -Year, names_to = "Month", values_to = "Value") |>
    mutate(Month = match(Month, month.abb), Month = as.Date(paste(Year, Month, "01",
        sep = "-")), ) |>
    dplyr::select(Month, Year, Value, )
long_Traffic <- long_Traffic |>
    mutate(Date = Month, .before = Month)
long_Traffic <- long_Traffic |>
    mutate(Month = lubridate::month(long_Traffic$Date))


# The number of cointegrating vectors depends on number of distinct BLS
# categories that you think are determined by the economic conditions of
# Larimer county.  These variables should go first in the dataframe df. They
# should be followed by the variables that are driven state or country-level
# conditions.  Then add additional data you want to add (licenses and/or data
# you think is cointegrated with the sales tax data).  Then add the sales tax
# data.  Log all variables
df <- cbind(FinalData, transportation_employment = long_Trans$transportation_employment,
    MAN_employment = long_Manf$MAN_employment, AFS_employment = long_Food$AFS_employment,
    RT_employment = long_Retail$RT_employment, long_Traffic$Value)
df <- df |>
    mutate(month = month(df$Month), .before = year)
df <- df |>
    dplyr::select(-Month)
new_df <- df |>
    dplyr::select("LH", "INFO", "EDUHS", "TTU", "FA", "MAN_employment", "AFS_employment",
        "RT_employment", "long_Traffic$Value", "Lead_STF_Real")
ldf <- log(new_df)
ldf <- ldf |>
    rename(traffic_frequency = `long_Traffic$Value`)
CombinedTS <- ts(ldf, start = c(2012, 1), frequency = 12)
# ts.plot(CombinedTS)
```

#========== # C. Create VECM model using data up to 2025/01 #==========

Johansen procedure

```
# Conduct the Johansen procedure
Johansen <- ca.jo(ldf, ecdet = "none", type = "eigen", K = 12)  # K = number of periods in a year (4 if
summary(Johansen)
```

```
##
## ######################
## # Johansen-Procedure #
## ######################
##
## Test type: maximal eigenvalue statistic (lambda max) , with linear trend
##
## Eigenvalues (lambda):
```

```
##  [1] 0.926047659 0.770387734 0.676282701 0.517538749 0.484404853 0.355292880
##  [7] 0.274540645 0.212601501 0.172308365 0.003635216
##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 9 |   0.52  6.50  8.18 11.65
## r <= 8 |  27.23 12.91 14.90 19.19
## r <= 7 |  34.42 18.90 21.07 25.75
## r <= 6 |  46.22 24.78 27.14 32.14
## r <= 5 |  63.21 30.84 33.32 38.78
## r <= 4 |  95.39 36.25 39.43 44.59
## r <= 3 | 104.96 42.06 44.91 51.30
## r <= 2 | 162.42 48.43 51.07 57.07
## r <= 1 | 211.88 54.01 57.00 63.37
## r = 0  | 375.02 59.00 62.42 68.61
##
## Eigenvectors, normalised to first column:
## (These are the cointegration relations)
##
##                            LH.l12   INFO.l12  EDUHS.l12    TTU.l12
## LH.l12                 1.000000000  1.0000000  1.0000000  1.0000000
## INFO.l12               0.098971466 -0.2538590  2.1219574  0.0881995
## EDUHS.l12             -0.310502114  3.3352171  7.6388916  1.9082511
## TTU.l12                0.966618291 -2.5590985 -4.2302417 -1.6442436
## FA.l12                -0.002014748 -0.6539578  1.5660543 -1.8706565
## MAN_employment.l12    -0.311095817 -3.1725604 -1.9882149 -3.9245208
## AFS_employment.l12    -0.961223928 -0.1004245  0.1933213 -2.0536687
## RT_employment.l12     -0.662373464  7.6868451 -3.5850978  4.6839915
## traffic_frequency.l12  0.133253873 -3.5590449 -8.7339932  0.3435247
## Lead_STF_Real.l12     -0.085828200 -0.1562862  0.5422054  2.1566201
##                            FA.l12 MAN_employment.l12 AFS_employment.l12
## LH.l12                 1.000000000         1.00000000         1.00000000
## INFO.l12              -0.114301081        -0.07275617        -0.22268479
## EDUHS.l12              0.004272198        -1.25959157        -1.77694814
## TTU.l12               -0.815812072         4.94821832        -2.60455885
## FA.l12                -0.067036419        -2.55142206         0.16608003
## MAN_employment.l12    -0.292147045        -0.99284603         4.74191027
## AFS_employment.l12    -0.907417055        -1.90795583        -2.07990219
## RT_employment.l12      1.935285209        -6.17297434         1.49751556
## traffic_frequency.l12 -0.520316029         4.54738615        -0.17618546
## Lead_STF_Real.l12      0.256486579         0.66211318        -0.05787494
##                       RT_employment.l12 traffic_frequency.l12 Lead_STF_Real.l12
## LH.l12                       1.00000000            1.00000000        1.00000000
## INFO.l12                    -0.03999662           -0.24640595       -0.00865867
## EDUHS.l12                    0.10185776            1.28172908       -1.07215438
## TTU.l12                     -2.75830330           -1.43750984        1.71983451
## FA.l12                       0.11763403            0.45170398        0.78182767
## MAN_employment.l12           1.34156623           -1.95238849        1.19577361
## AFS_employment.l12          -1.06543541           -1.61816909       -1.47777266
## RT_employment.l12            3.05071502            2.39064032       -2.57144054
## traffic_frequency.l12       -0.94376758            0.05315751        0.82261044
## Lead_STF_Real.l12           -0.06244587            0.57003719       -1.09355808
##
```

```
## Weights W:
## (This is the loading matrix)
##
##                            LH.l12    INFO.l12    EDUHS.l12      TTU.l12    FA.l12
## LH.d                   -4.71169568 -0.58728995 -0.118370326  0.264007012 0.7223611
## INFO.d                 -2.52844742 -0.20696077 -0.016593526 -0.094564459 0.1240612
## EDUHS.d                -0.85037390 -0.17664745 -0.026051885  0.040121575 0.5213256
## TTU.d                   0.14780616 -0.20213641 -0.008338594  0.078254114 0.1918824
## FA.d                   -1.23766504 -0.08766505  0.012387882  0.064612254 0.2711052
## MAN_employment.d        0.08451316 -0.11561829 -0.010697180  0.003362671 0.0141350
## AFS_employment.d       -3.47592309 -0.48348673 -0.115592926  0.291844835 0.9589312
## RT_employment.d        -0.14781263 -0.27901976 -0.022852306  0.096065081 0.4178895
## traffic_frequency.d    -1.34538210 -0.45400104 -0.032783140 -0.186598039 1.1380222
## Lead_STF_Real.d         2.57103243 -0.04260720  0.043039053 -0.244802923 0.5796221
##                       MAN_employment.l12 AFS_employment.l12 RT_employment.l12
## LH.d                          0.433611905       -0.092557463       -0.69845141
## INFO.d                        0.151713156       -0.216593117        0.01381981
## EDUHS.d                       0.080757755        0.004989062       -0.26075275
## TTU.d                         0.081738048       -0.018922970       -0.07803927
## FA.d                         -0.001809863       -0.129110971       -0.12270511
## MAN_employment.d              0.015060992       -0.104583327       -0.12165959
## AFS_employment.d              0.395464154       -0.129617442       -0.62284888
## RT_employment.d               0.135047546       -0.047513636       -0.13678868
## traffic_frequency.d          -0.081178665        0.004434832       -0.26034610
## Lead_STF_Real.d               0.341918107        0.089038578       -0.73207654
##                       traffic_frequency.l12 Lead_STF_Real.l12
## LH.d                              0.331322986       0.052103172
## INFO.d                           -0.029445123       0.028442028
## EDUHS.d                           0.003809199       0.016288311
## TTU.d                             0.076429993       0.011723897
## FA.d                              0.032545572       0.008231735
## MAN_employment.d                  0.039065922       0.003489427
## AFS_employment.d                  0.319655755       0.053144978
## RT_employment.d                   0.114412483       0.010916167
## traffic_frequency.d               0.524063812       0.038593639
## Lead_STF_Real.d                   0.272705440       0.033877265
```

Confirm the number of cointegrating vectors based on the results of the Johansen procedure.

Print the matrix of cointegrating vectors.

```
Beta <- round(coefB(Johansen, r = 9), 3)  # r = number of cointegrating vectors
Beta
```

```
##                          [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]
## LH.l12                1.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
## INFO.l12              0.000  1.000  0.000  0.000  0.000  0.000  0.000  0.000
## EDUHS.l12             0.000  0.000  1.000  0.000  0.000  0.000  0.000  0.000
## TTU.l12               0.000  0.000  0.000  1.000  0.000  0.000  0.000  0.000
## FA.l12                0.000  0.000  0.000  0.000  1.000  0.000  0.000  0.000
## MAN_employment.l12    0.000  0.000  0.000  0.000  0.000  1.000  0.000  0.000
## AFS_employment.l12    0.000  0.000  0.000  0.000  0.000  0.000  1.000  0.000
## RT_employment.l12     0.000  0.000  0.000  0.000  0.000  0.000  0.000  1.000
## traffic_frequency.l12 0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.000
## Lead_STF_Real.l12    -0.861 -0.542 -0.953 -0.666 -0.601 -0.795 -0.775 -0.395
##                          [,9]
```

```
## LH.l12                 0.000
## INFO.l12               0.000
## EDUHS.l12              0.000
## TTU.l12                0.000
## FA.l12                 0.000
## MAN_employment.l12     0.000
## AFS_employment.l12     0.000
## RT_employment.l12      0.000
## traffic_frequency.l12  1.000
## Lead_STF_Real.l12     -0.584
```

Short run adjustment coefficients

```
vecm <- VECM(ldf, lag = 11, r = 9, estim = "ML", include = "const")  # lag = K-1
# summary(vecm)
```

For each vector, we need at least one adjustment speed to have the oppositive sign of the corresponding cointegrating vector coefficient and be statistically significant.

#========== # D. Test if model predicts 2024 data well #==========

Rerun the model on data up to 2024/01

```
# truncate the data by removing last 12 observations
numrow <- nrow(ldf)
ldf2023 <- ldf[1:(numrow - 12), ]
# Fit the same VECM model
Johansen2023 <- ca.jo(ldf2023, ecdet = "none", type = "eigen", K = 12)
summary(Johansen2023)  # Johansen using log'd data (2023)
```

```
##
## ######################
## # Johansen-Procedure #
## ######################
##
## Test type: maximal eigenvalue statistic (lambda max) , with linear trend
##
## Eigenvalues (lambda):
##  [1] 0.99514533 0.98392263 0.89119685 0.78874813 0.66781834 0.60130715
##  [7] 0.32428793 0.27103414 0.14522750 0.06037026
##
## Values of teststatistic and critical values of test:
##
##           test 10pct  5pct  1pct
## r <= 9 |   8.22  6.50  8.18 11.65
## r <= 8 |  20.71 12.91 14.90 19.19
## r <= 7 |  41.73 18.90 21.07 25.75
## r <= 6 |  51.74 24.78 27.14 32.14
## r <= 5 | 121.38 30.84 33.32 38.78
## r <= 4 | 145.47 36.25 39.43 44.59
## r <= 3 | 205.22 42.06 44.91 51.30
## r <= 2 | 292.80 48.43 51.07 57.07
## r <= 1 | 545.21 54.01 57.00 63.37
## r = 0  | 703.27 59.00 62.42 68.61
##
## Eigenvectors, normalised to first column:
```

```
## (These are the cointegration relations)
##
##                            LH.l12    INFO.l12  EDUHS.l12     TTU.l12
## LH.l12                  1.0000000  1.00000000  1.0000000  1.00000000
## INFO.l12                0.3420458  0.17676328  0.0437246  0.06009702
## EDUHS.l12              -0.3448651 -0.50524684  0.5932711  0.09026252
## TTU.l12                 1.1541694  1.74525236 -0.2411361  0.12274812
## FA.l12                  0.4787557 -0.02901567 -0.4302576 -0.17131798
## MAN_employment.l12      0.2730978 -0.52567513 -1.2437121 -0.44933652
## AFS_employment.l12     -0.9473584 -0.95821563 -0.7172257 -0.84099566
## RT_employment.l12      -1.0975663 -1.39618431  1.7648448  0.12455281
## traffic_frequency.l12 -0.6880350  0.35164866 -0.7910198 -0.11055899
## Lead_STF_Real.l12      -0.5654354 -0.09793089  0.3782381  0.14363714
##                                FA.l12 MAN_employment.l12 AFS_employment.l12
## LH.l12                  1.0000000000          1.0000000         1.00000000
## INFO.l12               -0.4268082887         -0.3588405        -0.01459594
## EDUHS.l12               1.4896155592         -2.4762509        -1.35125560
## TTU.l12                -5.5263903580         -1.0257297        -0.01659233
## FA.l12                  1.7502227692          0.9266696        -0.34957920
## MAN_employment.l12      1.6772703115          1.1806133         2.79482853
## AFS_employment.l12     -0.0007336483         -1.4569775        -1.29690150
## RT_employment.l12       7.7981308442          4.2137875        -1.70031797
## traffic_frequency.l12 -4.4059818394         -0.7080184         0.69815655
## Lead_STF_Real.l12      -1.3830920909          0.3668607        -0.19999191
##                       RT_employment.l12 traffic_frequency.l12 Lead_STF_Real.l12
## LH.l12                        1.0000000            1.00000000       1.000000000
## INFO.l12                      0.6511206           -0.02190141      -0.206646664
## EDUHS.l12                    -1.0903987           -0.00573747       0.542507504
## TTU.l12                       9.4095554            0.15168288      -1.216404726
## FA.l12                       -1.2382337           -0.03332791      -0.261046315
## MAN_employment.l12           -2.7098734           -0.05757721      -0.118661295
## AFS_employment.l12            0.1195931           -1.03610156      -0.885103125
## RT_employment.l12           -12.5533341           -0.23487270       1.497410773
## traffic_frequency.l12         3.3444500            0.09587093      -0.208970819
## Lead_STF_Real.l12            -0.2788431           -0.01684577       0.005542403
##
## Weights W:
## (This is the loading matrix)
##
##                              LH.l12    INFO.l12  EDUHS.l12     TTU.l12
## LH.d                  -1.4256037168 -0.43296835 -3.9907379 -7.23716778
## INFO.d                -0.2880577660 -1.53612671 -2.4185758 -2.22810392
## EDUHS.d               -0.3774844929 -0.44586722 -1.4030315 -1.44255124
## TTU.d                 -0.0514046167 -0.07052456 -1.0212846 -0.66809592
## FA.d                   0.0169047731 -1.51876487 -0.6719324 -0.08709898
## MAN_employment.d      -0.1244883837  0.08271394 -0.8541998 -0.10343150
## AFS_employment.d      -1.2593128100  0.49754317 -3.7067851 -7.01607691
## RT_employment.d       -0.2547105345  0.15783352 -1.5299097 -1.02034858
## traffic_frequency.d -0.6091242533 -1.32470644 -5.4127009 -3.23010168
## Lead_STF_Real.d       -0.0007782574 -0.64914569 -1.7174793 -8.63418411
##                           FA.l12 MAN_employment.l12 AFS_employment.l12
## LH.d                  -0.83768689       -0.233434569         0.06832485
## INFO.d                 0.01418346       -0.066154721        -0.06549944
## EDUHS.d               -0.18230720        0.063617998         0.09201896
```

```
## TTU.d               -0.21150823       -0.013810729        -0.03912522
## FA.d                -0.11423407        0.001755664        -0.28574278
## MAN_employment.d    -0.04500721       -0.019779564        -0.11744662
## AFS_employment.d    -0.80190426       -0.172061444        -0.05578849
## RT_employment.d     -0.31030736        0.005367405        -0.05696425
## traffic_frequency.d  0.22560090        0.146077953        -0.24726448
## Lead_STF_Real.d     -0.16560136        0.056538443        -0.07000478
##                     RT_employment.l12 traffic_frequency.l12 Lead_STF_Real.l12
## LH.d                      0.043082676          -0.90062368       -1.02957902
## INFO.d                   -0.001777403          -0.92089810        0.14904360
## EDUHS.d                   0.059815643          -0.08683749       -0.02573729
## TTU.d                    -0.062607660          -0.03420695       -0.20427019
## FA.d                      0.013577958           0.17865074       -0.06149604
## MAN_employment.d          0.069470162          -0.10863242       -0.09418147
## AFS_employment.d         -0.008944553          -0.78834967       -0.92195287
## RT_employment.d          -0.063087861          -0.27365129       -0.38486005
## traffic_frequency.d       0.038384713           1.46197443       -0.94658523
## Lead_STF_Real.d           0.271437027          -0.63005294       -0.46685713
```

Forecast 2024 data

```
# vec2var: Transform a VECM to VAR object in levels
vec2var_ca.jo2023 <- vec2var(Johansen2023, r = 8)
# forecasting horizon (12 months)
nhor <- 12
# Forecasting for 2024
pred_vec2var_ca.jo2024 <- predict(vec2var_ca.jo2023, n.ahead = nhor)
```

Compare actual and forecasted 2024 lead_STF_Real Calculate Mean Absolute Error for 2024

```
# Forecast2024 <- pred_vec2var_ca.jo2024$fcst$FinalData.lead_STF_Real[,1] #
# returns NULL
Forecast2024 <- pred_vec2var_ca.jo2024[["fcst"]][["Lead_STF_Real"]][, 1]  # imported manually from RStu
Actual2024 <- FinalData[(nrow(FinalData) - 11):nrow(FinalData), "Lead_STF_Real"]

AE <- abs(Forecast2024 - Actual2024)
MAE <- mean(AE[, 1])

Forecast2024 <- pred_vec2var_ca.jo2024[["fcst"]][["Lead_STF_Real"]][, 1]  # Predicts log(Lead_STF_Real)
Forecast2024_expo <- exp(Forecast2024)  # Exponentiation of the forecast to undo log() function

Actual2024 <- ldf$Lead_STF_Real[FinalData$year == 2024]
Actual2024_expo <- exp(Actual2024)  # Exponentiation of the observations to undo log() function

round(Actual2024, 4) == round(ldf$Lead_STF_Real[FinalData$year == 2024], 4)
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
round(Actual2024_expo, 4) == round(FinalData$Lead_STF_Real[FinalData$year == 2024],
    4)
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
MAE
```

```
## [1] 3.34572
```

```r
mae <- mean(AE)
AE_expo <- abs(Forecast2024_expo - Actual2024_expo)
MAE_expo <- mean(AE_expo)
round(MAE_expo, 5)
```

```
## [1] 0.00974
```

You can redo steps A-D many times until you find the model with the lowest MAE. Play with the variables included in your model and with the number of months included in your dataset. Try including data starting in 2013, or in 2014, or later, instead of 2012, and see if you get better predictions.

#========== # E. Forecast lead_STF_Real for the months 2/25 to 11/25 and save it. #==========

```r
# vec2var: Transform a VECM to VAR in levels This is the original model, using
# all the data
vec2var_ca.jo <- vec2var(Johansen, r = 9)
# forecasting horizon
nhor <- 10
# Forecasting 2025 (10 observations)
pred_vec2var_ca.jo <- predict(vec2var_ca.jo, n.ahead = nhor)
Forecast2025 <- as.data.frame(pred_vec2var_ca.jo[["fcst"]][["Lead_STF_Real"]][, 1])
Forecast2025 <- Forecast2025 |>
    rename(Log_Lead_STF_Real_2025 = `pred_vec2var_ca.jo[["fcst"]][["Lead_STF_Real"]][, 1]`)

# One column for log(sales tax) and a column for reversing log()
Forecast2025 <- Forecast2025 |>
    mutate(Lead_STF_Real_2025 = exp(Log_Lead_STF_Real_2025))
# Undo adjustment

Forecast2025$Log_Lead_STF_Real_2025 <- Forecast2025$Log_Lead_STF_Real_2025 * (4.35/3.85)
Forecast2025$Lead_STF_Real_2025 <- Forecast2025$Lead_STF_Real_2025 * (4.35/3.85)


print(Forecast2025)
```

```
##    Log_Lead_STF_Real_2025 Lead_STF_Real_2025
## 1               -3.417640         0.05487459
## 2               -3.543063         0.04910907
## 3               -3.086300         0.07357497
## 4               -2.961747         0.08214959
## 5               -3.117279         0.07158509
## 6               -2.939966         0.08374854
## 7               -2.912889         0.08577986
## 8               -2.966172         0.08182846
## 9               -2.920442         0.08520834
## 10              -3.036781         0.07687126
```

```r
write_csv(Forecast2025, "Data/Forecast2025.csv")
```