

Lab 5

Abdulla Almansoori, Alexander Firth, Nathan Seto

2025-04-07

```
#===== # 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$Lead_STF_Real <- FinalData$Lead_STF_Real * (3.85/4.25)  
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] 0.926047626 0.770387915 0.676282705 0.517538748 0.484404892 0.355292929
## [7] 0.274540644 0.212601529 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.112  INFO.112  EDUHS.112  TTU.112
## LH.112          1.00000000  1.0000000  1.0000000  1.00000000
## INFO.112         0.09897149 -0.2538563  2.1219351  0.08819922
## EDUHS.112        -0.31050238  3.3351918  7.6387958  1.90824909
## TTU.112          0.96661878 -2.5590793 -4.2301898 -1.64424388
## FA.112           -0.00201476 -0.6539523  1.5660388 -1.87065445
## MAN_employment.112 -0.31109573 -3.1725381 -1.9881865 -3.92451709
## AFS_employment.112 -0.96122399 -0.1004308  0.1933054 -2.05366753
## RT_employment.112  -0.66237424  7.6867899 -3.5850701  4.68399034
## traffic_frequency.112 0.13325424 -3.5590209 -8.7338891  0.34352357
## Lead_STF_Real.112  -0.08582821 -0.1562847  0.5422015  2.15661840
##
##          FA.112  MAN_employment.112  AFS_employment.112
## LH.112          1.000000000  1.00000000  1.000000000
## INFO.112         -0.114301046  -0.07275637  -0.22268477
## EDUHS.112         0.004271929  -1.25959149  -1.77694809
## TTU.112          -0.815812379  4.94822485  -2.60455847
## FA.112           -0.067036022  -2.55142477  0.16607992
## MAN_employment.112 -0.292146429  -0.99285065  4.74190991
## AFS_employment.112 -0.907416972  -1.90795604  -2.07990215
## RT_employment.112  1.935285193  -6.17297986  1.49751523
## traffic_frequency.112 -0.520316252  4.54738999  -0.17618527
## Lead_STF_Real.112  0.256486376  0.66211420  -0.05787491
##
##          RT_employment.112  traffic_frequency.112  Lead_STF_Real.112
## LH.112          1.00000000  1.0000000  1.00000000
## INFO.112         -0.03999662  -0.2464060  -0.00865867
## EDUHS.112         0.10185769  1.2817293  -1.07215437
## TTU.112          -2.75830277  -1.4375094  1.71983449
## FA.112           0.11763388  0.4517040  0.78182767
## MAN_employment.112  1.34156599  -1.9523893  1.19577362
## AFS_employment.112 -1.06543552  -1.6181693  -1.47777266
## RT_employment.112  3.05071431  2.3906400  -2.57144052
## traffic_frequency.112 -0.94376710  0.0531579  0.82261043
## Lead_STF_Real.112  -0.06244577  0.5700374  -1.09355808
##
## Weights W:
## (This is the loading matrix)

```

```
##
##          LH.112      INFO.112      EDUHS.112      TTU.112      FA.112
## LH.d      -4.71168967 -0.58729519 -0.118371573  0.26400740 0.72236098
## INFO.d     -2.52844550 -0.20696261 -0.016593696 -0.09456452 0.12406137
## EDUHS.d    -0.85037240 -0.17664893 -0.026052150  0.04012167 0.52132577
## TTU.d       0.14780764 -0.20213800 -0.008338658  0.07825422 0.19188250
## FA.d       -1.23766450 -0.08766573  0.012388030  0.06461234 0.27110515
## MAN_employment.d  0.08451406 -0.11561923 -0.010697285  0.00336269 0.01413515
## AFS_employment.d -3.47591806 -0.48349105 -0.115594147  0.29184525 0.95893108
## RT_employment.d  -0.14781047 -0.27902200 -0.022852520  0.09606523 0.41788958
## traffic_frequency.d -1.34537929 -0.45400466 -0.032783441 -0.18659808 1.13802344
## Lead_STF_Real.d   2.57103262 -0.04260736  0.043039530 -0.24480318 0.57962186
##
##          MAN_employment.112 AFS_employment.112 RT_employment.112
## LH.d      0.433611605      -0.092557335      -0.69845061
## INFO.d     0.151713021      -0.216593102      0.01382006
## EDUHS.d    0.080757731      0.004989089      -0.26075263
## TTU.d      0.081737971      -0.018922945      -0.07803908
## FA.d       -0.001809812      -0.129110974      -0.12270516
## MAN_employment.d  0.015060969      -0.104583333      -0.12165950
## AFS_employment.d  0.395463912      -0.129617318      -0.62284818
## RT_employment.d  0.135047438      -0.047513596      -0.13678840
## traffic_frequency.d -0.081178597      0.004434794      -0.26034595
## Lead_STF_Real.d   0.341917884      0.089038639      -0.73207617
##
##          traffic_frequency.112 Lead_STF_Real.112
## LH.d      0.33132306      0.052103048
## INFO.d    -0.02944509      0.028441988
## EDUHS.d    0.00380924      0.016288285
## TTU.d      0.07643000      0.011723876
## FA.d       0.03254557      0.008231738
## MAN_employment.d  0.03906593      0.003489413
## AFS_employment.d  0.31965582      0.053144875
## RT_employment.d  0.11441250      0.010916133
## traffic_frequency.d  0.52406371      0.038593616
## Lead_STF_Real.d   0.27270546      0.033877241
```

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.112      1.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## INFO.112     0.000 1.000 0.000 0.000 0.000 0.000 0.000 0.000
## EDUHS.112    0.000 0.000 1.000 0.000 0.000 0.000 0.000 0.000
## TTU.112      0.000 0.000 0.000 1.000 0.000 0.000 0.000 0.000
## FA.112       0.000 0.000 0.000 0.000 1.000 0.000 0.000 0.000
## MAN_employment.112 0.000 0.000 0.000 0.000 0.000 1.000 0.000 0.000
## AFS_employment.112 0.000 0.000 0.000 0.000 0.000 0.000 1.000 0.000
## RT_employment.112 0.000 0.000 0.000 0.000 0.000 0.000 0.000 1.000
## traffic_frequency.112 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## Lead_STF_Real.112 -0.861 -0.542 -0.953 -0.666 -0.601 -0.795 -0.775 -0.395
##          [,9]
## LH.112      0.000
## INFO.112     0.000
```

```
## EDUHS.112          0.000
## TTU.112            0.000
## FA.112             0.000
## MAN_employment.112 0.000
## AFS_employment.112 0.000
## RT_employment.112  0.000
## traffic_frequency.112 1.000
## Lead_STF_Real.112  -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 opposite 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] 0.99514534 0.98392291 0.89119752 0.78874791 0.66781856 0.60130715
## [7] 0.32428799 0.27103428 0.14522748 0.06037027
##
## 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 |  292.81 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.112    INFO.112    EDUHS.112    TTU.112
## LH.112      1.0000000    1.00000000    1.00000000    1.00000000
## INFO.112     0.3420557    0.17676539    0.04372561    0.06009722
## EDUHS.112   -0.3448547   -0.50525621    0.59326249    0.09026210
## TTU.112     1.1541356    1.74527150   -0.24112107    0.12274996
## FA.112      0.4787852   -0.02901122   -0.43025325   -0.17131771
## MAN_employment.112  0.2731417   -0.52566829   -1.24370631   -0.44933701
## AFS_employment.112 -0.9473573   -0.95821761   -0.71722746   -0.84099566
## RT_employment.112 -1.0975456   -1.39621297    1.76482149    0.12455137
## traffic_frequency.112 -0.6880977    0.35165546   -0.79101165   -0.11055879
## Lead_STF_Real.112  -0.5654624   -0.09793628    0.37823355    0.14363680
##          FA.112 MAN_employment.112 AFS_employment.112
## LH.112      1.00000000000    1.0000000    1.000000000
## INFO.112    -0.4268132680    -0.3588370    -0.01459513
## EDUHS.112     1.4896278956    -2.4762290    -1.35125393
## TTU.112     -5.5264500408    -1.0257220    -0.01658339
## FA.112      1.7502490058     0.9266612    -0.34957993
## MAN_employment.112  1.6772976193    1.1805988    2.79482025
## AFS_employment.112 -0.0007252096    -1.4569716    -1.29689950
## RT_employment.112  7.7982082547     4.2137605    -1.70032574
## traffic_frequency.112 -4.4060267502    -0.7080175     0.69815746
## Lead_STF_Real.112  -1.3831126924     0.3668586    -0.19999159
##          RT_employment.112 traffic_frequency.112 Lead_STF_Real.112
## LH.112      1.0000000    1.0000000000    1.000000000
## INFO.112     0.6511215    -0.021901408    -0.206646637
## EDUHS.112    -1.0903959    -0.005737459     0.542507412
## TTU.112      9.4095669     0.151682662    -1.216404476
## FA.112     -1.2382343    -0.033327872    -0.261046308
## MAN_employment.112  -2.7098851    -0.057577178    -0.118661286
## AFS_employment.112  0.1195956    -1.036101576    -0.885103130
## RT_employment.112  -12.5533441    -0.234872308     1.497410442
## traffic_frequency.112  3.3444511     0.095870769    -0.208970741
## Lead_STF_Real.112  -0.2788427    -0.016845740     0.005542393
##
## Weights W:
## (This is the loading matrix)
##
##          LH.112    INFO.112    EDUHS.112    TTU.112    FA.112
## LH.d      -1.4255194053   -0.43298914   -3.9907858   -7.2371816   -0.83767562
## INFO.d     -0.2880358050   -1.53610681   -2.4186071   -2.2281097    0.01418399
## EDUHS.d    -0.3774611359   -0.44586796   -1.4030498   -1.4425563   -0.18230476
## TTU.d      -0.0514016059   -0.07051404   -1.0212971   -0.6680979   -0.21150545
## FA.d       0.0169089806   -1.51874692   -0.6719492   -0.0871033   -0.11423271
## MAN_employment.d -0.1244815689    0.08271561   -0.8542077   -0.1034318   -0.04500642
## AFS_employment.d -1.2592414497    0.49752244   -3.7068242   -7.0160877   -0.80189338
## RT_employment.d -0.2546965532    0.15783757   -1.5299264   -1.0203514   -0.31030325
## traffic_frequency.d -0.6090847952   -1.32467618   -5.4127572   -3.2301092    0.22559977
## Lead_STF_Real.d -0.0007757916   -0.64911124   -1.7175026   -8.6341940   -0.16559875
##          MAN_employment.112 AFS_employment.112 RT_employment.112
## LH.d      -0.233437510     0.06832168     0.043081747
## INFO.d     -0.066155530     -0.06550061    -0.001777619
## EDUHS.d     0.063618241     0.09201842     0.059815321
## TTU.d      -0.013811039     -0.03912612    -0.062607741
## FA.d       0.001755570      -0.28574356     0.013578148

```

```
## MAN_employment.d      -0.019779810      -0.11744698      0.069470091
## AFS_employment.d      -0.172063831      -0.05579186      -0.008945317
## RT_employment.d       0.005367147      -0.05696555      -0.063088031
## traffic_frequency.d    0.146078679      -0.24726663      0.038384599
## Lead_STF_Real.d       0.056538298      -0.07000592      0.271436381
##                        traffic_frequency.l12 Lead_STF_Real.l12
## LH.d                  -0.90063092      -1.02957970
## INFO.d                -0.92090079      0.14904329
## EDUHS.d               -0.08683932      -0.02573734
## TTU.d                 -0.03420799      -0.20427028
## FA.d                  0.17864974      -0.06149639
## MAN_employment.d      -0.10863325      -0.09418146
## AFS_employment.d      -0.78835618      -0.92195336
## RT_employment.d       -0.27365295      -0.38486016
## traffic_frequency.d    1.46197042      -0.94658536
## Lead_STF_Real.d       -0.63005749      -0.46685695
```

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] 3.440148
```

```
mae <- mean(AE)
AE_expo <- abs(Forecast2024_expo - Actual2024_expo)
MAE_expo <- mean(AE_expo)
```

```
round(MAE_expo, 5)
```

```
## [1] 0.00882
```

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. #=====
```

```
# 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 * (4.25/3.85)  
Forecast2025$Lead_STF_Real_2025 <- Forecast2025$Lead_STF_Real_2025 * (4.25/3.85)  
print(Forecast2025)
```

```
##      Log_Lead_STF_Real_2025 Lead_STF_Real_2025  
## 1          -3.448184          0.04856739  
## 2          -3.570722          0.04346460  
## 3          -3.124460          0.06511847  
## 4          -3.002768          0.07270767  
## 5          -3.154726          0.06335732  
## 6          -2.981489          0.07412277  
## 7          -2.955035          0.07592055  
## 8          -3.007093          0.07242333  
## 9          -2.962416          0.07541456  
## 10         -3.076085          0.06803554
```

```
# write_csv(Forecast2025, 'Data/Forecast2025.csv')
```