```
1  **Appendix A.1
2
3  * Zhi Zheng
4  * Time Series Modeling and Forecasting
5  * Spring 2022
6  * Project AllEmployees
7
8  clear
9  set more off
10
11 *set the working directory to wherever you have the data"
12 cd "E:\Time Series\Project"
13
14 *create log
15 log using "Project_1", replace
16
17 *Import data from csv file.
18 import delimited "Project.csv"
19
20 *Check to make sure data is imported.
21 describe
22 summarize
23
24 *Rename the four variables.
25
26 *Average Weekly Earnings of All Employees: Total Private in Miami-Fort Lauderdale-West Palm Beach,
   FL (MSA)
27 rename smu12331000500000011 WeeklyWage
28
29 *Average Hourly Earnings of All Employees: Total Private in Miami-Fort Lauderdale-West Palm Beach,
   FL (MSA)
30 rename smu12331000500000003 HourlyWage
31
32 *Average Weekly Hours of All Employees: Total Private in Miami-Fort Lauderdale-West Palm Beach, FL
   (MSA)
33 rename smu12331000500000002 WeeklyHrs
34
35 *All Employees: Total Private in Miami-Fort Lauderdale-West Palm Beach, FL (MSA)
36 rename smu12331000500000001 AllEmployees
37
38 *Civilian Labor Force in Miami-Fort Lauderdale-West Palm Beach, FL (MSA)
39 rename  miam112lfn Labor_Force
40
41 *Unemployment Rate in Miami-Fort Lauderdale-West Palm Beach, FL (MSA)
42 rename miam112urn Unem_Rate
43
44 *Generate a monthly date variable (make its display format monthly time, %tm)
45 generate datestring=date(date,"YMD")
46 gen datec = mofd(datestring)
47 format datec %tm
48 tsset datec
49
50 keep if tin(2007m1,2022m2)
51
52 *add January 2020 to the data,
53 tsappend, add(1)
54 gen month=month(dofm(datec))
55
56 *Generate dummy month indicators
57 tabulate month, generate(m)
58
59
60 *Generate natural logs of the variables to be used in the analysis
```

```
gen lnLabF=ln(Labor_Force)

gen lnUnemRate=ln(Unem_Rate)

gen lnAllEmployees=ln(AllEmployees)

gen lnWeeklyWage=ln(WeeklyWage)

gen lnHourlyWage=ln(HourlyWage)

gen lnWeeklyHrs=ln(WeeklyHrs)

/*
*tsline plots
tsline lnLabF, title("tsline lnLabF") saving("tsline1", replace)

tsline lnUnemRate, title("tsline lnUnemRate") saving("tsline2", replace)

tsline lnAllEmployees, title("tsline lnAllEmployees") saving("tsline3", replace)

graph combine "tsline1" "tsline2" "tsline3", rows(2)
graph export "tsline1.emf", replace

*AC
ac lnLabF, title("Autocorrelogram lnLabF") saving("ac1", replace)

ac lnUnemRate, title("Autocorrelogram lnUnemRate") saving("ac2", replace)

ac lnAllEmployees, title("Autocorrelogram lnAllEmployees") saving("ac3", replace)

graph combine "ac1" "ac2" "ac3", rows(2)
graph export "dependence1.emf", replace


*PAC
pac lnLabF, title("Partial Autocorrelogram lnLabF") saving("pac1", replace)

pac lnUnemRate, title("Partial Autocorrelogram lnUnemRate") saving("pac2", replace)

pac lnAllEmployees, title("Partial Autocorrelogram lnAllEmployees") saving("pac3", replace)

graph combine "pac1" "pac2" "pac3", rows(2)
graph export "dependence2.emf", replace


*Generate lags for vselect
gen dlnAllEmployees = d.lnAllEmployees

quietly forvalues i = 1/12 {
    gen dlnAllEmployeesl`i' = l`i'd.lnAllEmployees
}

quietly forvalues i = 1/12 {
    gen dlnLabFl`i'= l`i'd.lnLabF
}

quietly forvalues i = 1/12 {
    gen dlnUnemRate`i'= l`i'd.lnUnemRate
}

*Vselecting the models for Total Private Employee
vselect dlnAllEmployees dlnAllEmployeesl* dlnLabFl* dlnUnemRate*, best fix(m2 m3 m4 m5 m6 m7 m8 m9
m10 m11 m12)
```

```
123
124 *Check LOOCV for them
125 scalar drop _all
126
127 reg d.lnAllEmployees l(1/12)d.lnAllEmployees m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
128     if tin(2008m1, 2022m2)
129 estat ic // getting ic
130         scalar define df0=el(r(S),1,4) // saving model df
131         scalar define aic0=el(r(S),1,5) // saving aic
132         scalar define bic0=el(r(S),1,6) // saving bic
133 loocv reg d.lnAllEmployees l(12)d.lnAllEmployees m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
134     if tin(2008m1, 2022m2)
135         scalar define loormse0=r(rmse)
136
137 reg d.lnAllEmployees l(1)d.lnAllEmployees l(1,8)d.lnUnemRate l(2)d.lnLabF m2 m3 m4 m5 m6 m7 m8 m9
    m10 m11 m12 ///
138     if tin(2008m1, 2022m2)
139 estat ic // getting ic
140         scalar define df1=el(r(S),1,4) // saving model df
141         scalar define aic1=el(r(S),1,5) // saving aic
142         scalar define bic1=el(r(S),1,6) // saving bic
143 loocv reg d.lnAllEmployees l(1)d.lnAllEmployees l(1,8)d.lnUnemRate l(2)d.lnLabF m2 m3 m4 m5 m6 m7 m8
    m9 m10 m11 m12 ///
144     if tin(2008m1, 2022m2)
145         scalar define loormse1=r(rmse)
146
147 reg d.lnAllEmployees l(1)d.lnAllEmployees l(1,8)d.lnUnemRate l(2, 6)d.lnLabF m2 m3 m4 m5 m6 m7 m8 m9
    m10 m11 m12 ///
148     if tin(2008m1, 2022m2)
149 estat ic // getting ic
150         scalar define df2=el(r(S),1,4) // saving model df
151         scalar define aic2=el(r(S),1,5) // saving aic
152         scalar define bic2=el(r(S),1,6) // saving bic
153 loocv reg d.lnAllEmployees l(1)d.lnAllEmployees l(1,8)d.lnUnemRate l(2, 6)d.lnLabF m2 m3 m4 m5 m6 m7
    m8 m9 m10 m11 m12 ///
154     if tin(2008m1, 2022m2)
155         scalar define loormse2=r(rmse)
156
157 reg d.lnAllEmployees l(1)d.lnAllEmployees l(1, 6, 8)d.lnUnemRate l(2, 5)d.lnLabF m2 m3 m4 m5 m6 m7
    m8 m9 m10 m11 m12 ///
158     if tin(2008m1, 2022m2)
159 estat ic // getting ic
160         scalar define df3=el(r(S),1,4) // saving model df
161         scalar define aic3=el(r(S),1,5) // saving aic
162         scalar define bic3=el(r(S),1,6) // saving bic
163 loocv reg d.lnAllEmployees l(1)d.lnAllEmployees l(1, 6, 8)d.lnUnemRate l(2, 5)d.lnLabF m2 m3 m4 m5
    m6 m7 m8 m9 m10 m11 m12 ///
164     if tin(2008m1, 2022m2)
165         scalar define loormse3=r(rmse)
166
167 reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 4)d.lnUnemRate l(5, 6)d.lnLabF m2 m3 m4 m5 m6
    m7 m8 m9 m10 m11 m12 ///
168     if tin(2008m1, 2022m2)
169 estat ic // getting ic
170         scalar define df4=el(r(S),1,4) // saving model df
171         scalar define aic4=el(r(S),1,5) // saving aic
172         scalar define bic4=el(r(S),1,6) // saving bic
173 loocv reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 4)d.lnUnemRate l(5, 6)d.lnLabF m2 m3 m4
    m5 m6 m7 m8 m9 m10 m11 m12 ///
174     if tin(2008m1, 2022m2)
175         scalar define loormse4=r(rmse)
176
177 reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 4, 9)d.lnUnemRate l(5, 6)d.lnLabF m2 m3 m4 m5
```

```
    m6 m7 m8 m9 m10 m11 m12 ///
178     if tin(2008m1, 2022m2)
179 estat ic // getting ic
180         scalar define df5=el(r(S),1,4) // saving model df
181         scalar define aic5=el(r(S),1,5) // saving aic
182         scalar define bic5=el(r(S),1,6) // saving bic
183 loocv reg d.lnAllEmployees l(1)d.lnAllEmployees l(1, 2, 6, 8, 9)d.lnUnemRate l(2, 5)d.lnLabF m2 m3
    m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
184     if tin(2008m1, 2022m2)
185         scalar define loormse5=r(rmse)
186
187 reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 2, 4, 6, 9)d.lnUnemRate l(5)d.lnLabF m2 m3 m4
    m5 m6 m7 m8 m9 m10 m11 m12 ///
188     if tin(2008m1, 2022m2)
189 estat ic // getting ic
190         scalar define df6=el(r(S),1,4) // saving model df
191         scalar define aic6=el(r(S),1,5) // saving aic
192         scalar define bic6=el(r(S),1,6) // saving bic
193 loocv reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 2, 6, 9)d.lnUnemRate l(5)d.lnLabF m2 m3
    m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
194     if tin(2008m1, 2022m2)
195         scalar define loormse6=r(rmse)
196
197 reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 2, 4, 6, 7, 9)d.lnUnemRate l(5)d.lnLabF m2 m3
    m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
198     if tin(2008m1, 2022m2)
199 estat ic // getting ic
200         scalar define df7=el(r(S),1,4) // saving model df
201         scalar define aic7=el(r(S),1,5) // saving aic
202         scalar define bic7=el(r(S),1,6) // saving bic
203 loocv reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 2, 4, 6, 7, 9)d.lnUnemRate l(5)d.lnLabF
    m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
204     if tin(2008m1, 2022m2)
205         scalar define loormse7=r(rmse)
206
207 *Creating a comparision table
208 matrix drop _all
209 matrix fit0=(df0,aic0,bic0,loormse0)
210 matrix fit1=(df1,aic1,bic1,loormse1)
211 matrix fit2=(df2,aic2,bic2,loormse2)
212 matrix fit3=(df3,aic3,bic3,loormse3)
213 matrix fit4=(df4,aic4,bic4,loormse4)
214 matrix fit5=(df5,aic5,bic5,loormse5)
215 matrix fit6=(df6,aic6,bic6,loormse6)
216 matrix fit7=(df7,aic7,bic7,loormse7)
217
218 matrix FIT=fit0\fit1\fit2\fit3\fit4\fit5\fit6\fit7
219 matrix rownames FIT= "Model 12-Lag AR" "Model 4" "Model 5" "Model 6" "Model 7" "Model 8" "Model 9"
    "Model 10"
220 matrix colnames FIT=df AIC BIC LOOCV
221 matrix list FIT
222
223 summ datec if l12d.lnAllEmployees~=. & l9d.lnUnemRate~=. & l6d.lnLabF~=.
224
225 summ datec if datec==tm(2022m2)
226
227 *Rolling window program
228 scalar drop _all
229 quietly forval w=48(12)180 {
230 /* w=small(inc)large
231 small is the smallest window
232 inc is the window size increment
233 large is the largest window.
```

```
(large-small)/inc must be an interger */
gen pred=. // out of sample prediction
gen nobs=. // number of observations in the window for each forecast point
    forval t=673/745 {
    /* t=first/last
    first is the first date for which you want to make a forecast.
    first-1 is the end date of the earliest window used to fit the model.
    first-w, where w is the window width, is the date of the first
    observation used to fit the model in the earliest window.
    You must choose first so it is preceded by a full set of
    lags for the model with the longest lag length to be estimated.
    last is  the last observation to be forecast. */
    gen wstart=`t'-`w' // fit window start date
    gen wend=`t'-1 // fit window end date
    /* Enter the regression command immediately below.
    Leave the if statement intact to control the window  */
    reg d.lnAllEmployees l(1/12)d.lnAllEmployees m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
        if datec>=wstart & datec<=wend // restricts the model to the window
    replace nobs=e(N) if datec==`t' // number of observations used
    predict ptemp // temporary predicted values
    replace pred=ptemp if datec==`t' // saving the single forecast value
    drop ptemp wstart wend // clear these to prepare for the next loop
    }
gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
summ errsq // getting the mean of the squared errors
scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
summ nobs // getting min and max obs used
scalar RWminobs`w'=r(min) // min obs used in the window width
scalar RWmaxobs`w'=r(max) // max obs used in the window width
drop errsq pred nobs // clearing for the next loop
}
scalar list // list the RMSE and min and max obs for each window width

*Rolling window program
scalar drop _all
quietly forval w=48(12)180 {
/* w=small(inc)large
small is the smallest window
inc is the window size increment
large is the largest window.
(large-small)/inc must be an interger */
gen pred=. // out of sample prediction
gen nobs=. // number of observations in the window for each forecast point
    forval t=673/745 {
    /* t=first/last
    first is the first date for which you want to make a forecast.
    first-1 is the end date of the earliest window used to fit the model.
    first-w, where w is the window width, is the date of the first
    observation used to fit the model in the earliest window.
    You must choose first so it is preceded by a full set of
    lags for the model with the longest lag length to be estimated.
    last is  the last observation to be forecast. */
    gen wstart=`t'-`w' // fit window start date
    gen wend=`t'-1 // fit window end date
    /* Enter the regression command immediately below.
    Leave the if statement intact to control the window  */
    reg d.lnAllEmployees l(1)d.lnAllEmployees l(1,8)d.lnUnemRate l(2, 6)d.lnLabF m2 m3 m4 m5 m6 m7
m8 m9 m10 m11 m12 ///
        if datec>=wstart & datec<=wend // restricts the model to the window
    replace nobs=e(N) if datec==`t' // number of observations used
    predict ptemp // temporary predicted values
    replace pred=ptemp if datec==`t' // saving the single forecast value
    drop ptemp wstart wend // clear these to prepare for the next loop
```

```
296       }
297   gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
298   summ errsq // getting the mean of the squared errors
299   scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
300   summ nobs // getting min and max obs used
301   scalar RWminobs`w'=r(min) // in obs used in the window width
302   scalar RWmaxobs`w'=r(max) // max obs used in the window width
303   drop errsq pred nobs // clearing for the next loop
304   }
305   scalar list // list the RMSE and min and max obs for each window width
306
307   *Rolling window program
308   scalar drop _all
309   quietly forval w=48(12)180 {
310   /* w=small(inc)large
311   small is the smallest window
312   inc is the window size increment
313   large is the largest window.
314   (large-small)/inc must be an interger */
315   gen pred=. // out of sample prediction
316   gen nobs=. // number of observations in the window for each forecast point
317       forval t=673/745 {
318       /* t=first/last
319       first is the first date for which you want to make a forecast.
320       first-1 is the end date of the earliest window used to fit the model.
321       first-w, where w is the window width, is the date of the first
322       observation used to fit the model in the earliest window.
323       You must choose first so it is preceded by a full set of
324       lags for the model with the longest lag length to be estimated.
325       last is  the last observation to be forecast. */
326       gen wstart=`t'-`w' // fit window start date
327       gen wend=`t'-1 // fit window end date
328       /* Enter the regression command immediately below.
329       Leave the if statement intact to control the window  */
330   reg d.lnAllEmployees l(1)d.lnAllEmployees l(1, 6, 8)d.lnUnemRate l(2, 5)d.lnLabF m2 m3 m4 m5 m6 m7
      m8 m9 m10 m11 m12 ///
331           if datec>=wstart & datec<=wend // restricts the model to the window
332       replace nobs=e(N) if datec==`t' // number of observations used
333       predict ptemp // temporary predicted values
334       replace pred=ptemp if datec==`t' // saving the single forecast value
335       drop ptemp wstart wend // clear these to prepare for the next loop
336       }
337   gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
338   summ errsq // getting the mean of the squared errors
339   scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
340   summ nobs // getting min and max obs used
341   scalar RWminobs`w'=r(min) // in obs used in the window width
342   scalar RWmaxobs`w'=r(max) // max obs used in the window width
343   drop errsq pred nobs // clearing for the next loop
344   }
345   scalar list // list the RMSE and min and max obs for each window width
346
347   *Rolling window program
348   scalar drop _all
349   quietly forval w=48(12)180 {
350   /* w=small(inc)large
351   small is the smallest window
352   inc is the window size increment
353   large is the largest window.
354   (large-small)/inc must be an interger */
355   gen pred=. // out of sample prediction
356   gen nobs=. // number of observations in the window for each forecast point
357       forval t=673/745 {
```

```
    /* t=first/last
    first is the first date for which you want to make a forecast.
    first-1 is the end date of the earliest window used to fit the model.
    first-w, where w is the window width, is the date of the first
    observation used to fit the model in the earliest window.
    You must choose first so it is preceded by a full set of
    lags for the model with the longest lag length to be estimated.
    last is  the last observation to be forecast. */
    gen wstart=`t'-`w' // fit window start date
    gen wend=`t'-1 // fit window end date
    /* Enter the regression command immediately below.
    Leave the if statement intact to control the window  */
    reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 4)d.lnUnemRate l(5, 6)d.lnLabF m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
        if datec>=wstart & datec<=wend // restricts the model to the window
    replace nobs=e(N) if datec==`t' // number of observations used
    predict ptemp // temporary predicted values
    replace pred=ptemp if datec==`t' // saving the single forecast value
    drop ptemp wstart wend // clear these to prepare for the next loop
    }
gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
summ errsq // getting the mean of the squared errors
scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
summ nobs // getting min and max obs used
scalar RWminobs`w'=r(min) // in obs used in the window width
scalar RWmaxobs`w'=r(max) // max obs used in the window width
drop errsq pred nobs // clearing for the next loop
}
scalar list // list the RMSE and min and max obs for each window width


*Rolling window program
scalar drop _all
quietly forval w=48(12)180 {
/* w=small(inc)large
small is the smallest window
inc is the window size increment
large is the largest window.
(large-small)/inc must be an interger */
gen pred=. // out of sample prediction
gen nobs=. // number of observations in the window for each forecast point
    forval t=673/745 {
    /* t=first/last
    first is the first date for which you want to make a forecast.
    first-1 is the end date of the earliest window used to fit the model.
    first-w, where w is the window width, is the date of the first
    observation used to fit the model in the earliest window.
    You must choose first so it is preceded by a full set of
    lags for the model with the longest lag length to be estimated.
    last is  the last observation to be forecast. */
    gen wstart=`t'-`w' // fit window start date
    gen wend=`t'-1 // fit window end date
    /* Enter the regression command immediately below.
    Leave the if statement intact to control the window  */
    reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 4, 9)d.lnUnemRate l(5, 6)d.lnLabF m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
        if datec>=wstart & datec<=wend // restricts the model to the window
    replace nobs=e(N) if datec==`t' // number of observations used
    predict ptemp // temporary predicted values
    replace pred=ptemp if datec==`t' // saving the single forecast value
    drop ptemp wstart wend // clear these to prepare for the next loop
    }
gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
```

```
summ errsq // getting the mean of the squared errors
scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
summ nobs // getting min and max obs used
scalar RWminobs`w'=r(min) // in obs used in the window width
scalar RWmaxobs`w'=r(max) // max obs used in the window width
drop errsq pred nobs // clearing for the next loop
}
scalar list // list the RMSE and min and max obs for each window width

/*
Model 12-Lag: RWrmse180 =  .03774197
Model 5: RWrmse180 =  .04004232
Model 6: RWrmse180 =  .04157896
Model 7: RWrmse180 =  .03921726
Model 8: RWrmse180 =  .03858189
*/

*/


*Rolling window program -- Inner Loop Only

*So, the obs to fit are now 493+180=581 to 745.

scalar drop _all
gen pred=. // out of sample prediction
gen nobs=. // number of observations in the window for each forecast point
    quietly forval t=673/745 {

    /* t=first/last
    first is the first date for which you want to make a forecast.
    first-1 is the end date of the earliest window used to fit the model.
    first-w, where w is the window width, is the date of the first
    observation used to fit the model in the earliest window.
    You must choose first so it is preceded by a full set of
    lags for the model with the longest lag length to be estimated.
    last is  the last observation to be forecast. */

    gen wstart=`t'-180 // fit window start date
    gen wend=`t'-1 // fit window end date

    /* Enter the regression command immediately below.
    Leave the if statement intact to control the window */
    reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 4, 9)d.lnUnemRate l(5, 6)d.lnLabF m2 m3 m4
m5 m6 m7 m8 m9 m10 m11 m12 ///
        if datec>=wstart & datec<=wend // restricts the model to the window
    replace nobs=e(N) if datec==`t' // number of observations used
    predict ptemp // temporary predicted values
    replace pred=ptemp if datec==`t' // saving the single forecast value
    drop ptemp wstart wend // clear these to prepare for the next loop
    }
**End of selected rolling window implementation

*Examine Error Distribution
gen res=d.lnAllEmployees-pred
hist res, frac normal saving(errhist, replace) scheme(s1mono)
swilk res
sktest res

/*Run model on last window of 180 months (15 years)
to get most recent predictions and forecast*/

reg d.lnAllEmployees l(1, 2, 7)d.lnAllEmployees l(1, 4, 9)d.lnUnemRate l(5, 6)d.lnLabF m2 m3 m4 m5 m6
```

```
 m7 m8 m9 m10 m11 m12 ///
    if tin(2008m2,2022m2)
predict pdlnAllEmployees if datec==tm(2022m3) // generate point forecast
replace pdlnAllEmployees=pred if datec<tm(2022m3)

*Normal Interval
gen ressq=res^2 // generating squared errors
summ ressq // getting the mean of the squared errors

*95% Interval
gen pAllEmployeesn=exp(pdlnAllEmployees+l.lnAllEmployees+0.5*r(mean))
gen ubAllEmployeesn=pAllEmployeesn*exp(1.96*r(mean)^0.5)
gen lbAllEmployeesn=pAllEmployeesn*exp(-1.96*r(mean)^0.5)

*90% Interval
gen ubAllEmployeesn90=pAllEmployeesn*exp(1.64*r(mean)^0.5)
gen lbAllEmployeesn90=pAllEmployeesn*exp(-1.64*r(mean)^0.5)

*99% Interval
gen ubAllEmployeesn99=pAllEmployeesn*exp(2.58*r(mean)^0.5)
gen lbAllEmployeesn99=pAllEmployeesn*exp(-2.58*r(mean)^0.5)

twoway (tsline ubAllEmployeesn lbAllEmployeesn pAllEmployeesn if tin(2015m1,2022m3)) ///
    (scatter AllEmployees datec if tin(2015m2,2020m1), ms(Oh) ) ///
    (scatter AllEmployees datec if tin(2020m2,2022m3), ms(T) ) , ///
    scheme(s1mono) title("AllEmployees") ///
    t2title("Rolling Window Forecast Interval (Normal)") legend(order(1 "ubAllEmployeesn1" ///
        2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
"pAllEmployeesn2" 7 "AllEmployees") holes(2) )
    graph save AllEmployeesn.gph, replace

twoway (tsline ubAllEmployeesn90 pAllEmployeesn lbAllEmployeesn90 if tin(2018m1,2022m3)) ///
 (scatter AllEmployees datec if tin(2018m1,2022m2), ms(+) ) ///
 (scatter pAllEmployees datec if tin(2020m3,2020m3), ms(oh) ) , ///
 scheme(s1mono) title("AllEmployees 90% (Normal)") legend(off)
graph save AllEmployeesn90.gph, replace

twoway (tsline ubAllEmployeesn99 pAllEmployeesn lbAllEmployeesn99 if tin(2018m1,2022m3)) ///
 (scatter AllEmployees datec if tin(2018m1,2022m2), ms(+) ) ///
 (scatter pAllEmployees datec if tin(2020m3,2020m3), ms(oh) ) , ///
 scheme(s1mono) title("AllEmployees 99% (Normal)") legend(off)
graph save AllEmployeesn99.gph, replace

*Empirical Interval
gen experr=exp(res)
summ experr // mean is the multiplicative correction factor

gen pAllEmployeese=r(mean)*exp(l.lnAllEmployees+pdlnAllEmployees)

*95% Interval
_pctile experr, percentile(2.5,97.5) //  corrections for the bounds
return list

gen lbAllEmployeese=pAllEmployeese*r(r1)
gen ubAllEmployeese=pAllEmployeese*r(r2)

*90% Interval
_pctile experr, percentile(5,95) //  corrections for the bounds
return list

gen lbAllEmployeese90=r(r1)*pAllEmployeese
gen ubAllEmployeese90=r(r2)*pAllEmployeese
```

```
*99% Interval
_pctile experr, percentile(.5,99.5) //  corrections for the bounds
return list

gen lbAllEmployeese99=r(r1)*pAllEmployeese
gen ubAllEmployeese99=r(r2)*pAllEmployeese

twoway (tsline ubAllEmployeese lbAllEmployeese pAllEmployeese if tin(2015m1,2022m3)) ///
    (scatter AllEmployees datec if tin(2015m1,2022m2), ms(Oh) ) ///
    (scatter AllEmployees datec if tin(2022m3,2022m3), ms(T) ) , ///
    scheme(s1mono) title("All Employees") ///
    t2title("Rolling Window Forecast Interval (Empirical)") legend(order(1 "ubAllEmployeesn1" ///
        2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
"pAllEmployeesn2" 7 "AllEmployees") holes(2) )
    graph save AllEmployeese.gph, replace

twoway (tsline ubAllEmployeese90 lbAllEmployeese90 pAllEmployeese if tin(2015m1,2022m3)) ///
    (scatter AllEmployees datec if tin(2015m1,2022m2), ms(Oh) ) ///
    (scatter AllEmployees datec if tin(2022m3,2022m3), ms(T) ) , ///
    scheme(s1mono) title("All Employees") ///
    t2title("Rolling Window Forecast Interval (Empirical)") legend(order(1 "ubAllEmployeesn1" ///
        2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
"pAllEmployeesn2" 7 "AllEmployees") holes(2) )
    graph save AllEmployeese90.gph, replace

twoway (tsline ubAllEmployeese99 lbAllEmployeese99 pAllEmployeese if tin(2015m1,2022m3)) ///
    (scatter AllEmployees datec if tin(2015m1,2022m2), ms(Oh) ) ///
    (scatter AllEmployees datec if tin(2022m3,2022m3), ms(T) ) , ///
    scheme(s1mono) title("All Employees") ///
    t2title("Rolling Window Forecast Interval (Empirical)") legend(order(1 "ubAllEmployeesn1" ///
        2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
"pAllEmployeesn2" 7 "AllEmployees") holes(2) )
    graph save AllEmployeese99.gph, replace

    *Compare normal and empirical bounds
twoway (scatter AllEmployees datec, ms(Oh) ) ///
    (tsline lbAllEmployeesn ubAllEmployeesn lbAllEmployeese ubAllEmployeese, ///
        lpattern( solid solid "-###" "-###") ///
    lcolor(gs8%40 gs8%40 gs1 gs1) ///
    lwidth(vthick vthick thick thick) ) ///
    if tin(2020m1,2022m3) , tline(`=scalar(break)') scheme(s1mono) ///
    ylabel( , grid) xlabel( , grid) ///
    title("Miami-Fort Lauderdale-West Palm Beach AllEmployees") ///
    t2title("95% Forecast Interval Comparison") ///
    legend(order(1 "Actual" ///
        2 "Normal Bounds" 4 "Empirical Bounds" ) holes(2) )
    graph save AllEmployeesCombined.gph, replace

twoway (scatter AllEmployees datec, ms(Oh) ) ///
    (tsline lbAllEmployeesn90 ubAllEmployeesn90 lbAllEmployeese90 ubAllEmployeese90, ///
        lpattern( solid solid "-###" "-###") ///
    lcolor(gs8%40 gs8%40 gs1 gs1) ///
    lwidth(vthick vthick thick thick) ) ///
    if tin(2020m1,2022m3) , tline(`=scalar(break)') scheme(s1mono) ///
    ylabel( , grid) xlabel( , grid) ///
    title("Miami-Fort Lauderdale-West Palm Beach AllEmployees") ///
    t2title("90% Forecast Interval Comparison") ///
    legend(order(1 "Actual" ///
        2 "Normal Bounds" 4 "Empirical Bounds" ) holes(2) )
    graph save AllEmployeesCombined90.gph, replace

twoway (scatter AllEmployees datec, ms(Oh) ) ///
    (tsline lbAllEmployeesn99 ubAllEmployeesn99 lbAllEmployeese99 ubAllEmployeese99, ///
```

```
602           lpattern( solid solid "-###" "-###") ///
603       lcolor(gs8%40 gs8%40 gs1 gs1) ///
604       lwidth(vthick vthick thick thick) ) ///
605       if tin(2020m1,2022m3) , tline(`=scalar(break)') scheme(s1mono) ///
606       ylabel( , grid) xlabel( , grid) ///
607       title("Miami-Fort Lauderdale-West Palm Beach AllEmployees") ///
608       t2title("99% Forecast Interval Comparison") ///
609       legend(order(1 "Actual" ///
610           2 "Normal Bounds" 4 "Empirical Bounds" ) holes(2) )
611       graph save AllEmployeesCombined99.gph, replace
612
613   list lbAllEmployeesn pAllEmployeesn ubAllEmployeesn lbAllEmployeese pAllEmployeese ubAllEmployeese if
       datec==tm(2022m3)
614
615   Stop
616
617
618
619
620
621
622
623
624
```