

```

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,
27 FL (MSA)
28 rename smu12331000500000011 WeeklyWage
29
30 *Average Hourly Earnings of All Employees: Total Private in Miami-Fort Lauderdale-West Palm Beach,
31 FL (MSA)
32 rename smu12331000500000003 HourlyWage
33
34 *Average Weekly Hours of All Employees: Total Private in Miami-Fort Lauderdale-West Palm Beach, FL
35 (MSA)
36 rename smu12331000500000002 WeeklyHrs
37
38 *All Employees: Total Private in Miami-Fort Lauderdale-West Palm Beach, FL (MSA)
39 rename smu12331000500000001 AllEmployees
40
41 *Civilian Labor Force in Miami-Fort Lauderdale-West Palm Beach, FL (MSA)
42 rename miam1121fn Labor_Force
43
44 *Unemployment Rate in Miami-Fort Lauderdale-West Palm Beach, FL (MSA)
45 rename miam112urn Unem_Rate
46
47 *Generate a monthly date variable (make its display format monthly time, %tm)
48 generate datestring=date(date,"YMD")
49 gen datec = mofd(datestring)
50 format datec %tm
51 tsset datec
52
53 keep if tin(2007m1,2022m2)
54
55 *add January 2020 to the data,
56 tsappend, add(1)
57 gen month=month(dofm(datec))
58
59 *Generate dummy month indicators
60 tabulate month, generate(m)
61
62 *Generate natural logs of the variables to be used in the analysis

```

```

61  gen lnLabF=ln(Labor_Force)
62
63  gen lnUnemRate=ln(Unem_Rate)
64
65  gen lnAllEmployees=ln(AllEmployees)
66
67  gen lnWeeklyWage=ln(WeeklyWage)
68
69  gen lnHourlyWage=ln(HourlyWage)
70
71  gen lnWeeklyHrs=ln(WeeklyHrs)
72
73  /*
74  *tsline plots
75  tsline lnLabF, title("tsline lnLabF") saving("tsline1", replace)
76
77  tsline lnUnemRate, title("tsline lnUnemRate") saving("tsline2", replace)
78
79  tsline lnAllEmployees, title("tsline lnAllEmployees") saving("tsline3", replace)
80
81  graph combine "tsline1" "tsline2" "tsline3", rows(2)
82  graph export "tsline1.emf", replace
83
84  *AC
85  ac lnLabF, title("Autocorrelogram lnLabF") saving("ac1", replace)
86
87  ac lnUnemRate, title("Autocorrelogram lnUnemRate") saving("ac2", replace)
88
89  ac lnAllEmployees, title("Autocorrelogram lnAllEmployees") saving("ac3", replace)
90
91  graph combine "ac1" "ac2" "ac3", rows(2)
92  graph export "dependence1.emf", replace
93
94
95  *PAC
96  pac lnLabF, title("Partial Autocorrelogram lnLabF") saving("pac1", replace)
97
98  pac lnUnemRate, title("Partial Autocorrelogram lnUnemRate") saving("pac2", replace)
99
100  pac lnAllEmployees, title("Partial Autocorrelogram lnAllEmployees") saving("pac3", replace)
101
102  graph combine "pac1" "pac2" "pac3", rows(2)
103  graph export "dependence2.emf", replace
104
105
106  *Generate lags for vselect
107  gen dlnAllEmployees = d.lnAllEmployees
108
109  quietly forvalues i = 1/12 {
110      gen dlnAllEmployeesl`i' = l`i'd.lnAllEmployees
111  }
112
113  quietly forvalues i = 1/12 {
114      gen dlnLabFl`i' = l`i'd.lnLabF
115  }
116
117  quietly forvalues i = 1/12 {
118      gen dlnUnemRate`i' = l`i'd.lnUnemRate
119  }
120
121  *Vselecting the models for Total Private Employee
122  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.

```

```

234 (large-small)/inc must be an interger */
235 gen pred=. // out of sample prediction
236 gen nobs=. // number of observations in the window for each forecast point
237 forval t=673/745 {
238     /* t=first/last
239     first is the first date for which you want to make a forecast.
240     first-1 is the end date of the earliest window used to fit the model.
241     first-w, where w is the window width, is the date of the first
242     observation used to fit the model in the earliest window.
243     You must choose first so it is preceded by a full set of
244     lags for the model with the longest lag length to be estimated.
245     last is the last observation to be forecast. */
246     gen wstart=`t'-'w' // fit window start date
247     gen wend=`t'-1 // fit window end date
248     /* Enter the regression command immediately below.
249     Leave the if statement intact to control the window */
250     reg d.lnAllEmployees l(1/12)d.lnAllEmployees m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12 ///
251     if datec>=wstart & datec<=wend // restricts the model to the window
252     replace nobs=e(N) if datec==`t' // number of observations used
253     predict ptemp // temporary predicted values
254     replace pred=ptemp if datec==`t' // saving the single forecast value
255     drop ptemp wstart wend // clear these to prepare for the next loop
256 }
257 gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
258 summ errsq // getting the mean of the squared errors
259 scalar RWrmsq`w'=r(mean)^.5 // getting the rmse for window width i
260 summ nobs // getting min and max obs used
261 scalar RWminobs`w'=r(min) // min obs used in the window width
262 scalar RWmaxobs`w'=r(max) // max obs used in the window width
263 drop errsq pred nobs // clearing for the next loop
264 }
265 scalar list // list the RMSE and min and max obs for each window width
266
267 *Rolling window program
268 scalar drop _all
269 quietly forval w=48(12)180 {
270     /* w=small(inc)large
271     small is the smallest window
272     inc is the window size increment
273     large is the largest window.
274     (large-small)/inc must be an interger */
275     gen pred=. // out of sample prediction
276     gen nobs=. // number of observations in the window for each forecast point
277     forval t=673/745 {
278         /* t=first/last
279         first is the first date for which you want to make a forecast.
280         first-1 is the end date of the earliest window used to fit the model.
281         first-w, where w is the window width, is the date of the first
282         observation used to fit the model in the earliest window.
283         You must choose first so it is preceded by a full set of
284         lags for the model with the longest lag length to be estimated.
285         last is the last observation to be forecast. */
286         gen wstart=`t'-'w' // fit window start date
287         gen wend=`t'-1 // fit window end date
288         /* Enter the regression command immediately below.
289         Leave the if statement intact to control the window */
290         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 ///
291         if datec>=wstart & datec<=wend // restricts the model to the window
292         replace nobs=e(N) if datec==`t' // number of observations used
293         predict ptemp // temporary predicted values
294         replace pred=ptemp if datec==`t' // saving the single forecast value
295         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 nobss // 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 nobss // 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 nobss=. // 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
331         m8 m9 m10 m11 m12 ///
332         if datec>=wstart & datec<=wend // restricts the model to the window
333         replace nobss=e(N) if datec==`t' // number of observations used
334         predict ptemp // temporary predicted values
335         replace pred=ptemp if datec==`t' // saving the single forecast value
336         drop ptemp wstart wend // clear these to prepare for the next loop
337     }
338     gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
339     summ errsq // getting the mean of the squared errors
340     scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
341     summ nobss // getting min and max obs used
342     scalar RWminobs`w'=r(min) // in obs used in the window width
343     scalar RWmaxobs`w'=r(max) // max obs used in the window width
344     drop errsq pred nobss // clearing for the next loop
345 }
346 scalar list // list the RMSE and min and max obs for each window width
347
348 *Rolling window program
349 scalar drop _all
350 quietly forval w=48(12)180 {
351     /* w=small(inc)large
352     small is the smallest window
353     inc is the window size increment
354     large is the largest window.
355     (large-small)/inc must be an interger */
356     gen pred=. // out of sample prediction
357     gen nobss=. // number of observations in the window for each forecast point
358     forval t=673/745 {

```



```

358 /* t=first/last
359 first is the first date for which you want to make a forecast.
360 first-1 is the end date of the earliest window used to fit the model.
361 first-w, where w is the window width, is the date of the first
362 observation used to fit the model in the earliest window.
363 You must choose first so it is preceded by a full set of
364 lags for the model with the longest lag length to be estimated.
365 last is the last observation to be forecast. */
366 gen wstart=`t'-'w' // fit window start date
367 gen wend=`t'-1 // fit window end date
368 /* Enter the regression command immediately below.
369 Leave the if statement intact to control the window */
370 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 ///
371 if datec>=wstart & datec<=wend // restricts the model to the window
372 replace nobse=e(N) if datec==`t' // number of observations used
373 predict ptemp // temporary predicted values
374 replace pred=ptemp if datec==`t' // saving the single forecast value
375 drop ptemp wstart wend // clear these to prepare for the next loop
376 }
377 gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors
378 summ errsq // getting the mean of the squared errors
379 scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
380 summ nobse // getting min and max obs used
381 scalar RWminobs`w'=r(min) // in obs used in the window width
382 scalar RWmaxobs`w'=r(max) // max obs used in the window width
383 drop errsq pred nobse // clearing for the next loop
384 }
385 scalar list // list the RMSE and min and max obs for each window width
386
387
388 *Rolling window program
389 scalar drop _all
390 quietly forval w=48(12)180 {
391 /* w=small(inc)large
392 small is the smallest window
393 inc is the window size increment
394 large is the largest window.
395 (large-small)/inc must be an interger */
396 gen pred=. // out of sample prediction
397 gen nobse=. // number of observations in the window for each forecast point
398 forval t=673/745 {
399 /* t=first/last
400 first is the first date for which you want to make a forecast.
401 first-1 is the end date of the earliest window used to fit the model.
402 first-w, where w is the window width, is the date of the first
403 observation used to fit the model in the earliest window.
404 You must choose first so it is preceded by a full set of
405 lags for the model with the longest lag length to be estimated.
406 last is the last observation to be forecast. */
407 gen wstart=`t'-'w' // fit window start date
408 gen wend=`t'-1 // fit window end date
409 /* Enter the regression command immediately below.
410 Leave the if statement intact to control the window */
411 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 ///
412 if datec>=wstart & datec<=wend // restricts the model to the window
413 replace nobse=e(N) if datec==`t' // number of observations used
414 predict ptemp // temporary predicted values
415 replace pred=ptemp if datec==`t' // saving the single forecast value
416 drop ptemp wstart wend // clear these to prepare for the next loop
417 }
418 gen errsq=(pred-d.lnAllEmployees)^2 // generating squared errors

```

```

419 summ errsq // getting the mean of the squared errors
420 scalar RWrmse`w'=r(mean)^.5 // getting the rmse for window width i
421 summ nobs // getting min and max obs used
422 scalar RWminobs`w'=r(min) // in obs used in the window width
423 scalar RWmaxobs`w'=r(max) // max obs used in the window width
424 drop errsq pred nobs // clearing for the next loop
425 }
426 scalar list // list the RMSE and min and max obs for each window width
427
428 /*
429 Model 12-Lag: RWrmse180 = .03774197
430 Model 5: RWrmse180 = .04004232
431 Model 6: RWrmse180 = .04157896
432 Model 7: RWrmse180 = .03921726
433 Model 8: RWrmse180 = .03858189
434 */
435
436 */
437
438
439 *Rolling window program -- Inner Loop Only
440
441 *So, the obs to fit are now 493+180=581 to 745.
442
443 scalar drop _all
444 gen pred=. // out of sample prediction
445 gen nobs=. // number of observations in the window for each forecast point
446 quietly forval t=673/745 {
447
448     /* t=first/last
449     first is the first date for which you want to make a forecast.
450     first-1 is the end date of the earliest window used to fit the model.
451     first-w, where w is the window width, is the date of the first
452     observation used to fit the model in the earliest window.
453     You must choose first so it is preceded by a full set of
454     lags for the model with the longest lag length to be estimated.
455     last is the last observation to be forecast. */
456
457     gen wstart=`t'-180 // fit window start date
458     gen wend=`t'-1 // fit window end date
459
460     /* Enter the regression command immediately below.
461     Leave the if statement intact to control the window */
462     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 ///
463         if datec>=wstart & datec<=wend // restricts the model to the window
464         replace nobs=e(N) if datec==`t' // number of observations used
465         predict ptemp // temporary predicted values
466         replace pred=ptemp if datec==`t' // saving the single forecast value
467         drop ptemp wstart wend // clear these to prepare for the next loop
468     }
469 **End of selected rolling window implementation
470
471 *Examine Error Distribution
472 gen res=d.lnAllEmployees-pred
473 hist res, frac normal saving(errhist, replace) scheme(s1mono)
474 swilk res
475 sktest res
476
477 /*Run model on last window of 180 months (15 years)
478 to get most recent predictions and forecast*/
479
480 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 ///
481     if tin(2008m2,2022m2)
482 predict pdlnAllEmployees if datec==tm(2022m3) // generate point forecast
483 replace pdlnAllEmployees=pred if datec<tm(2022m3)
484
485 *Normal Interval
486 gen ressq=res^2 // generating squared errors
487 summ ressq // getting the mean of the squared errors
488
489 *95% Interval
490 gen pAllEmployeesn=exp(pdlnAllEmployees+1.lnAllEmployees+0.5*r(mean))
491 gen ubAllEmployeesn=pAllEmployeesn*exp(1.96*r(mean)^0.5)
492 gen lbAllEmployeesn=pAllEmployeesn*exp(-1.96*r(mean)^0.5)
493
494 *90% Interval
495 gen ubAllEmployeesn90=pAllEmployeesn*exp(1.64*r(mean)^0.5)
496 gen lbAllEmployeesn90=pAllEmployeesn*exp(-1.64*r(mean)^0.5)
497
498 *99% Interval
499 gen ubAllEmployeesn99=pAllEmployeesn*exp(2.58*r(mean)^0.5)
500 gen lbAllEmployeesn99=pAllEmployeesn*exp(-2.58*r(mean)^0.5)
501
502 twoway (tsline ubAllEmployeesn lbAllEmployeesn pAllEmployeesn if tin(2015m1,2022m3)) ///
503       (scatter AllEmployees datec if tin(2015m2,2020m1), ms(Oh) ) ///
504       (scatter AllEmployees datec if tin(2020m2,2022m3), ms(T) ) , ///
505       scheme(s1mono) title("AllEmployees") ///
506       t2title("Rolling Window Forecast Interval (Normal)") legend(order(1 "ubAllEmployeesn1" ///
507       2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
508       "pAllEmployeesn2" 7 "AllEmployees") holes(2) )
509 graph save AllEmployeesn.gph, replace
510
511 twoway (tsline ubAllEmployeesn90 pAllEmployeesn lbAllEmployeesn90 if tin(2018m1,2022m3)) ///
512       (scatter AllEmployees datec if tin(2018m1,2022m2), ms(+) ) ///
513       (scatter pAllEmployees datec if tin(2020m3,2020m3), ms(oh) ) , ///
514       scheme(s1mono) title("AllEmployees 90% (Normal)") legend(off)
515 graph save AllEmployeesn90.gph, replace
516
517 twoway (tsline ubAllEmployeesn99 pAllEmployeesn lbAllEmployeesn99 if tin(2018m1,2022m3)) ///
518       (scatter AllEmployees datec if tin(2018m1,2022m2), ms(+) ) ///
519       (scatter pAllEmployees datec if tin(2020m3,2020m3), ms(oh) ) , ///
520       scheme(s1mono) title("AllEmployees 99% (Normal)") legend(off)
521 graph save AllEmployeesn99.gph, replace
522
523 *Empirical Interval
524 gen experr=exp(res)
525 summ experr // mean is the multiplicative correction factor
526
527 gen pAllEmployeeese=r(mean)*exp(1.lnAllEmployees+pdlnAllEmployees)
528
529 *95% Interval
530 _pctile experr, percentile(2.5,97.5) // corrections for the bounds
531 return list
532
533 gen lbAllEmployeeese=pAllEmployeeese*r(r1)
534 gen ubAllEmployeeese=pAllEmployeeese*r(r2)
535
536 *90% Interval
537 _pctile experr, percentile(5,95) // corrections for the bounds
538 return list
539
540 gen lbAllEmployeeese90=r(r1)*pAllEmployeeese
541 gen ubAllEmployeeese90=r(r2)*pAllEmployeeese

```

```

542 *99% Interval
543 _ptile experr, percentile(.5,99.5) // corrections for the bounds
544 return list
545
546 gen lbAllEmployeeese99=r(r1)*pAllEmployeeese
547 gen ubAllEmployeeese99=r(r2)*pAllEmployeeese
548
549 twoway (tsline ubAllEmployeeese lbAllEmployeeese pAllEmployeeese if tin(2015m1,2022m3)) ///
550 (scatter AllEmployees datec if tin(2015m1,2022m2), ms(Oh) ) ///
551 (scatter AllEmployees datec if tin(2022m3,2022m3), ms(T) ) , ///
552 scheme(s1mono) title("All Employees") ///
553 t2title("Rolling Window Forecast Interval (Empirical)") legend(order(1 "ubAllEmployeesn1" ///
554 2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
"pAllEmployeesn2" 7 "AllEmployees") holes(2) )
555 graph save AllEmployeeese.gph, replace
556
557 twoway (tsline ubAllEmployeeese90 lbAllEmployeeese90 pAllEmployeeese if tin(2015m1,2022m3)) ///
558 (scatter AllEmployees datec if tin(2015m1,2022m2), ms(Oh) ) ///
559 (scatter AllEmployees datec if tin(2022m3,2022m3), ms(T) ) , ///
560 scheme(s1mono) title("All Employees") ///
561 t2title("Rolling Window Forecast Interval (Empirical)") legend(order(1 "ubAllEmployeesn1" ///
562 2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
"pAllEmployeesn2" 7 "AllEmployees") holes(2) )
563 graph save AllEmployeeese90.gph, replace
564
565 twoway (tsline ubAllEmployeeese99 lbAllEmployeeese99 pAllEmployeeese if tin(2015m1,2022m3)) ///
566 (scatter AllEmployees datec if tin(2015m1,2022m2), ms(Oh) ) ///
567 (scatter AllEmployees datec if tin(2022m3,2022m3), ms(T) ) , ///
568 scheme(s1mono) title("All Employees") ///
569 t2title("Rolling Window Forecast Interval (Empirical)") legend(order(1 "ubAllEmployeesn1" ///
570 2 "lbAllEmployeesn1" 3 "pAllEmployeesn1" 4 "ubAllEmployeesn2" 5 "lbAllEmployeesn2" 6
"pAllEmployeesn2" 7 "AllEmployees") holes(2) )
571 graph save AllEmployeeese99.gph, replace
572
573 *Compare normal and empirical bounds
574 twoway (scatter AllEmployees datec, ms(Oh) ) ///
575 (tsline lbAllEmployeesn ubAllEmployeesn lbAllEmployeeese ubAllEmployeeese, ///
576 lpattern( solid solid "-###" "-###") ///
577 lcolor(gs8%40 gs8%40 gs1 gs1) ///
578 lwidth(vthick vthick thick thick) ) ///
579 if tin(2020m1,2022m3) , tline(`=scalar(break)') scheme(s1mono) ///
580 ylabel( , grid) xlabel( , grid) ///
581 title("Miami-Fort Lauderdale-West Palm Beach AllEmployees") ///
582 t2title("95% Forecast Interval Comparison") ///
583 legend(order(1 "Actual" ///
584 2 "Normal Bounds" 4 "Empirical Bounds" ) holes(2) )
585 graph save AllEmployeesCombined.gph, replace
586
587 twoway (scatter AllEmployees datec, ms(Oh) ) ///
588 (tsline lbAllEmployeesn90 ubAllEmployeesn90 lbAllEmployeeese90 ubAllEmployeeese90, ///
589 lpattern( solid solid "-###" "-###") ///
590 lcolor(gs8%40 gs8%40 gs1 gs1) ///
591 lwidth(vthick vthick thick thick) ) ///
592 if tin(2020m1,2022m3) , tline(`=scalar(break)') scheme(s1mono) ///
593 ylabel( , grid) xlabel( , grid) ///
594 title("Miami-Fort Lauderdale-West Palm Beach AllEmployees") ///
595 t2title("90% Forecast Interval Comparison") ///
596 legend(order(1 "Actual" ///
597 2 "Normal Bounds" 4 "Empirical Bounds" ) holes(2) )
598 graph save AllEmployeesCombined90.gph, replace
599
600 twoway (scatter AllEmployees datec, ms(Oh) ) ///
601 (tsline lbAllEmployeesn99 ubAllEmployeesn99 lbAllEmployeeese99 ubAllEmployeeese99, ///

```

```
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 lbAllEmployeeese pAllEmployeeese ubAllEmployeeese if
        datec==tm(2022m3)
614
615     Stop
616
617
618
619
620
621
622
623
624
```