

Problem Set 5

Gus Lipkin

CAP 4763 Time Series Modelling and Forecasting

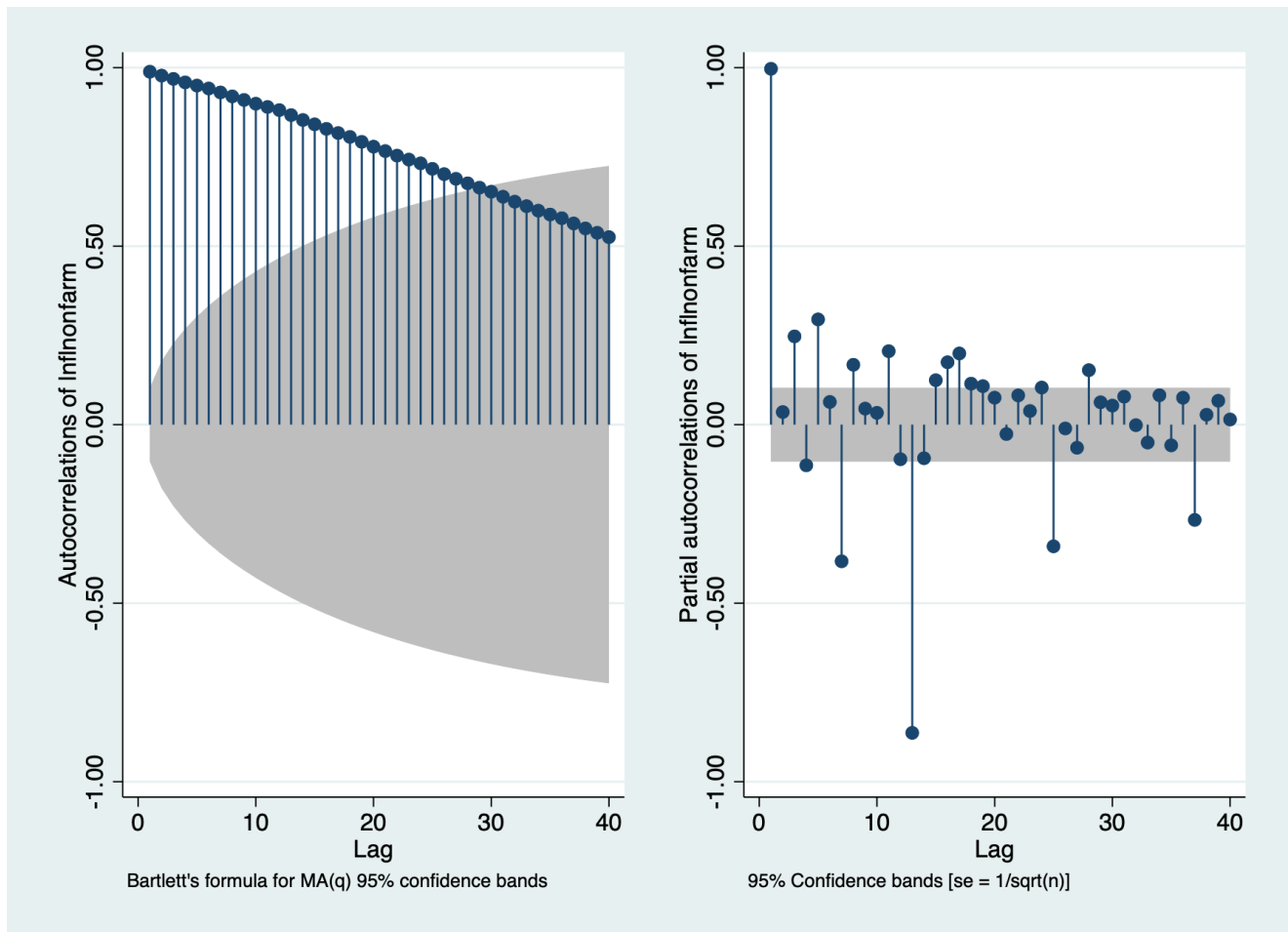
Table of Contents

Section
<u>Introduction</u>
<u>GSREG, Rolling Window, and Choosing Models</u>
<u>GSREG</u>
<u>Rolling Window</u>
<u>Using the Best Model to Forecast January 2020</u>
<u>Illustrations and Interpretations of the Models</u>
<u>Conclusion</u>
<u>Appendix A</u>
<u>Appendix B</u>

Introduction

Time Series Modelling and Forecasting allows us to be better prepared for the future by using past data to predict future trends. In this case, we are trying to predict total nonfarm employment for the state of Florida for January 2020.

First, we create log transforms of all the variables. Because the variables cannot be negative, conducting a log transform ensures that they will not become negative later on. Next, we can generate monthly indicators so that if data trends are tied to a specific month or are affected seasonally, the indicator values will show the correlations. The last step before conducting the time series modelling and forecasting is to determine the dataset's stationarity in time. To do this, we construct an autocorrelogram and partial-autocorrelogram.



The high partial autocorrelation of the first lag indicates that we should use the first difference of the data. This corrects for stationarity.

GSREG, Rolling Window, and Choosing Models

GSREG

Because GSREG runs through every possible combination of variables fed to it up to a maximum number of variables per regression, it is necessary to limit the number of variables. How these variables are chosen is up to the person running the analysis. When I ran mine, I decided to include the first, third, sixth, ninth, twelfth, and twenty-fourth lags of each differenced variable for `lnflnonfarm`, `lnfl1f`, and `lnusepr`. I also fixed the monthly indicators for January, March, June, and September. I chose to include all variables because while one variable may not have as heavy an influence, it is important to consider everything and conduct an analysis before dismissing any variables. I chose the lags and monthly indicators I did because while the data is monthly, I wanted to reduce the amount of variations that GSREG needs to go through without removing too many data points and without keeping too many variables that would cause the command to take too long to run. As for the twenty-fourth lag, I thought that there was a chance that long-term change would provide a grounding-point for the model so that it does not diverge too much. I understand that this can cause issues in times of immediate and rapid change such as the onset of COVID-19 but such events are few and far between.

The resulting best model suggested by GSREG is `reg d.lnflnonfarm l3d.lnflnonfarm l6d.lnflnonfarm l12d.lnflnonfarm l24d.lnflnonfarm l24d.lnfl1f l6d.lnusepr m1 m3 m6 m9`. The models with the least variables had the four fixed month indicators and two other variables. The best of these is `reg d.lnflnonfarm l12d.lnflnonfarm m1 m3 m6 m9`. The third model I chose was of an average length at eight variables: `reg d.lnflnonfarm l3d.lnflnonfarm l12d.lnflnonfarm l24d.lnflnonfarm l24d.lnflnonfarm l6d.lnusepr m1 m3 m6 m9`.

Rolling Window

Running each model against the whole dataset in a Rolling Window model has a Root Mean Square Error of .00388844, .00423688, and .00406403, respectively. The first and third models had window widths of 108 observations while the second had a window width of 120. With the lowest RMSE, I decided to move forward with the first model. Calculating the percentiles for 2.5 and 97.5 of the distribution gives $-.0074653569608927$ and .0065394379198551.

Using the Best Model to Forecast January 2020

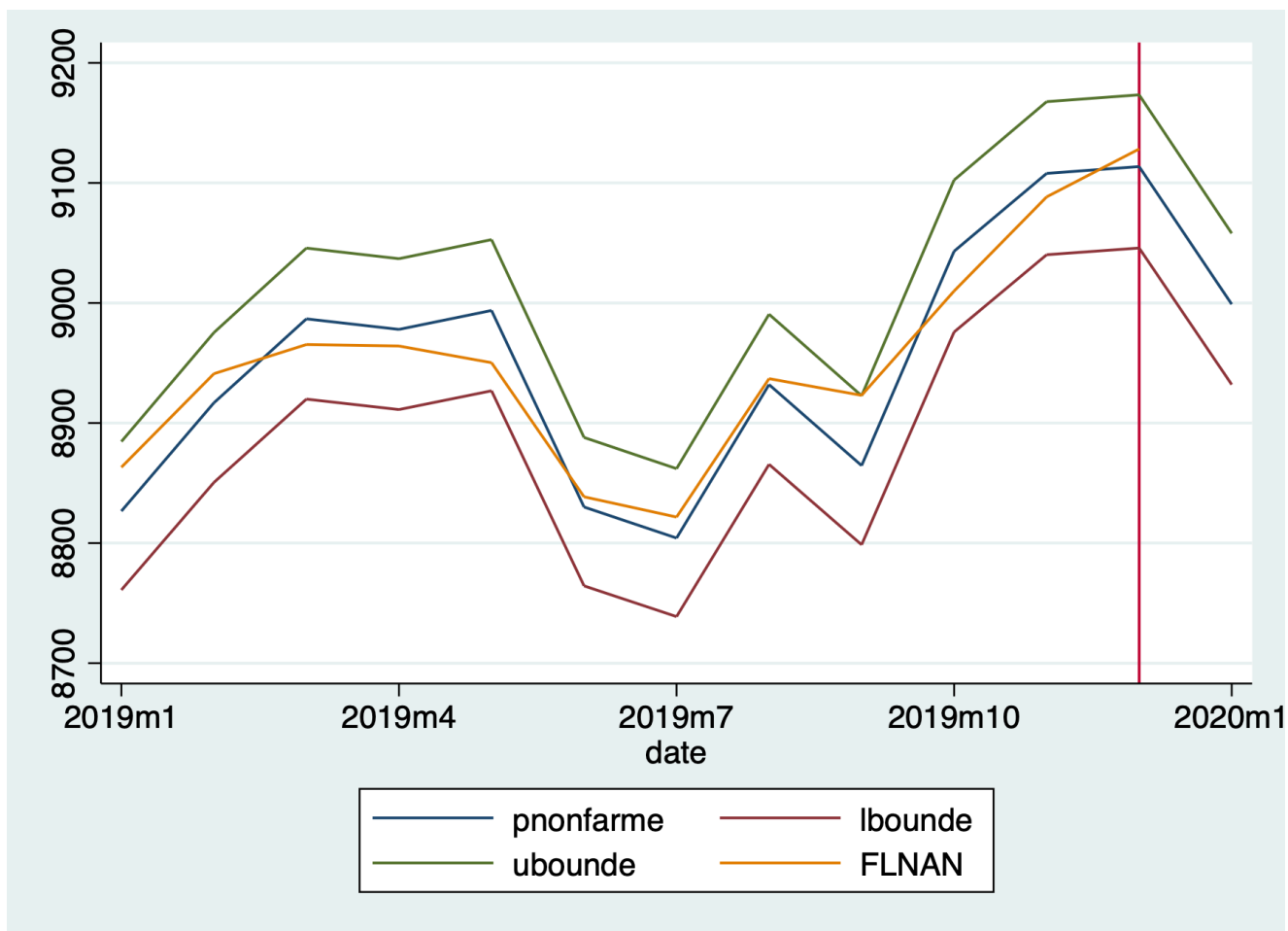
We can then use our best model and window width to forecast. If we limit our model data to several data points in the past, we can forecast the most recent past datapoints to check our model's accuracy.

month	pnonfarm	lbound	ubound
1	9001.077	8933.007	9069.667

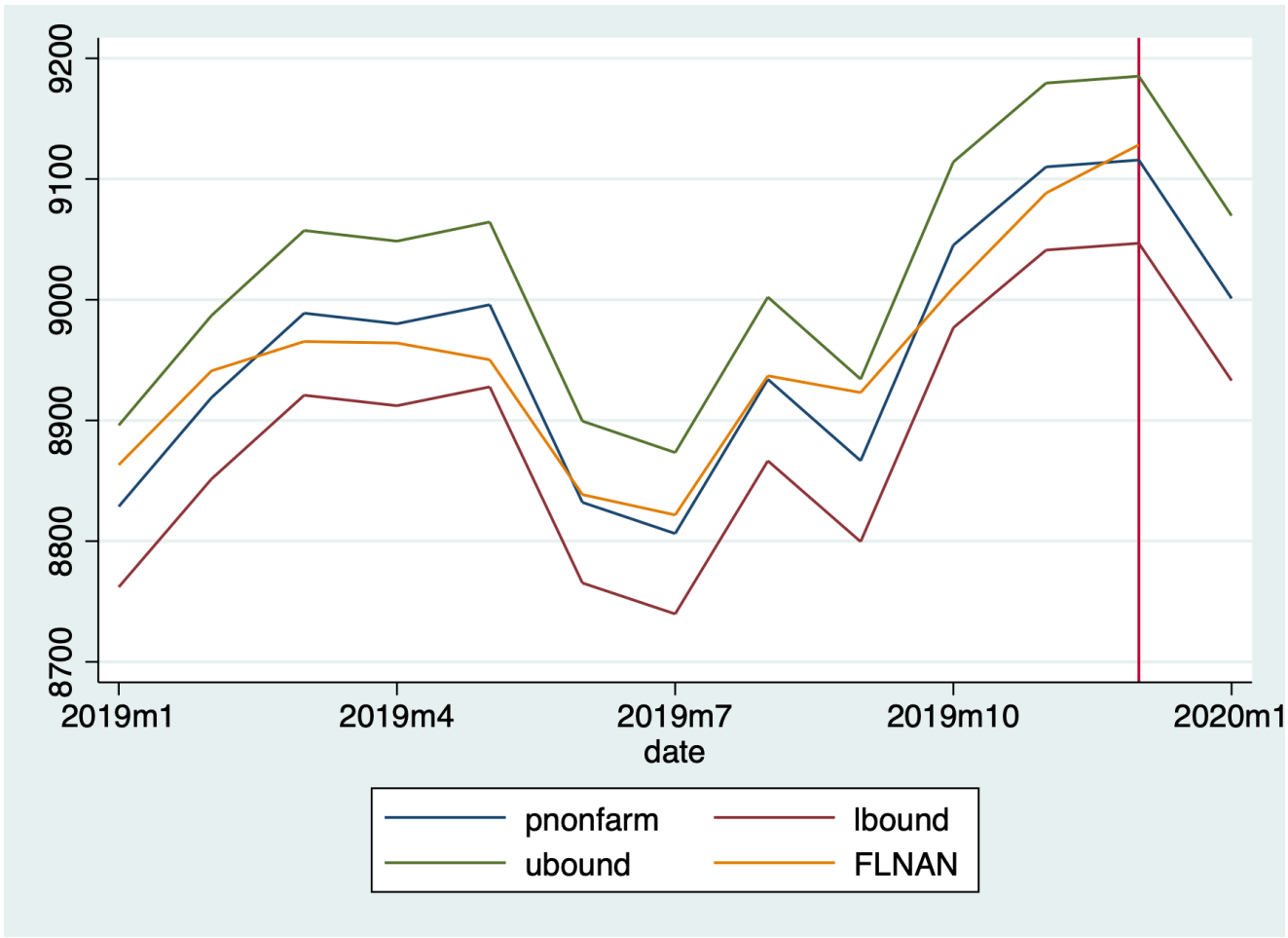
Table 1: One Month Ahead Forecast Predicting Non-farm Employment for January 2020

When we consult the actual data for January 2020, we see that nonfarm employment was at 9050.4. While our forecast is low, it is still within the bounds set.

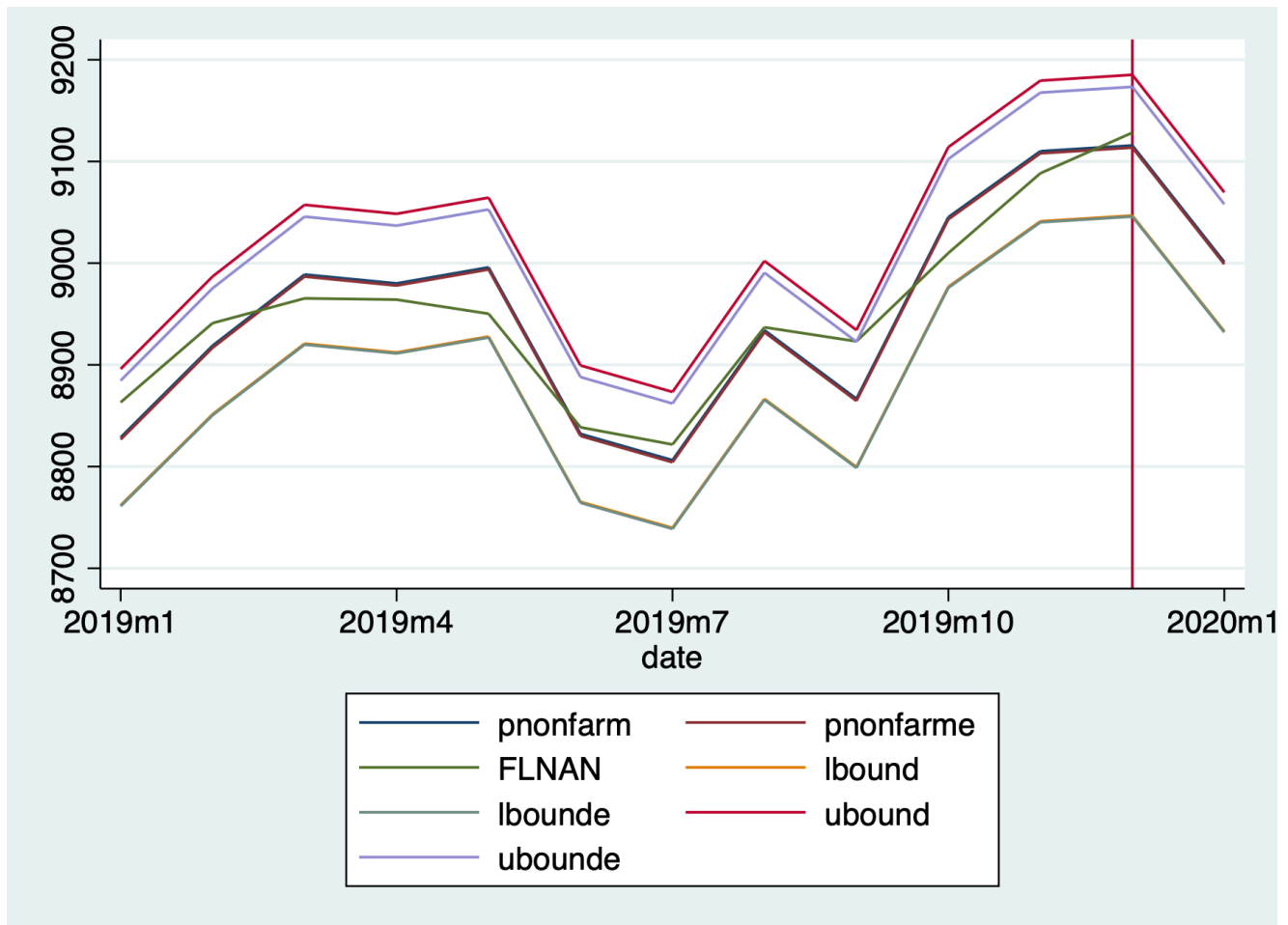
Illustrations and Interpretations of the Models



Nonfarm Empirical



Nonfarm Normal



Nonfarm Normal vs Empirical

Conclusion

With proper adjustment of variables through log transforms, dummy variables, and differencing, we can create time series models of the past and use it to forecast the future with relative accuracy leading to potential insights or predictions about the future.

Appendix A (Code)

```
1  clear
2  set more off
3
4  cd "/Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
   Sets/Problem Set 5"
5  log using "Problem Set 5", replace
6  import delimited "Assignment_1_Monthly.txt"
7
8  rename lnu02300000 us_epr
9  rename flnan fl_nonfarm
10 rename flllfn fl_lf
11 rename flbppriv fl_bp
12 rename date datestring
13
14 gen datec=date(datestring, "YMD")
15 gen date=mofd(datec)
16 gen month=month(datec)
17 format date %tm
18
19 tsset date
20
21 gen lnusepr=log(us_epr)
22 gen lnflnonfarm=log(fl_nonfarm)
23 gen lnflllf=log(fl_lf)
24 gen lnflbp=log(fl_bp)
25
26 *1
27 drop if !tin(1990m1,2019m12)
28
29 *2
30 tsset date
31 tsappend, add(1)
32 replace month=month(dofm(date)) if month==.
33
34 *interlude
35 ac lnflnonfarm, saving("ac_lnflnonfarm.gph", replace)
36 pac lnflnonfarm, saving("pac_lnflnonfarm.gph", replace)
37 graph combine ac_lnflnonfarm.gph pac_lnflnonfarm.gph
38 graph export ac_pac_lnflnonfarm.png, replace
39
40 *3
41 gen m1=0
42 replace m1=1 if month==1
43 gen m2=0
44 replace m2=1 if month==2
45 gen m3=0
46 replace m3=1 if month==3
```

```
47 gen m4=0
48 replace m4=1 if month==4
49 gen m5=0
50 replace m5=1 if month==5
51 gen m6=0
52 replace m6=1 if month==6
53 gen m7=0
54 replace m7=1 if month==7
55 gen m8=0
56 replace m8=1 if month==8
57 gen m9=0
58 replace m9=1 if month==9
59 gen m10=0
60 replace m10=1 if month==10
61 gen m11=0
62 replace m11=1 if month==11
63
64 gen dlnflnonfarm=d.lnflnonfarm
65 gen l1dlnflnonfarm=l1d.lnflnonfarm
66 gen l2dlnflnonfarm=l2d.lnflnonfarm
67 gen l3dlnflnonfarm=l3d.lnflnonfarm
68 gen l4dlnflnonfarm=l4d.lnflnonfarm
69 gen l5dlnflnonfarm=l5d.lnflnonfarm
70 gen l6dlnflnonfarm=l6d.lnflnonfarm
71 gen l7dlnflnonfarm=l7d.lnflnonfarm
72 gen l8dlnflnonfarm=l8d.lnflnonfarm
73 gen l9dlnflnonfarm=l9d.lnflnonfarm
74 gen l10dlnflnonfarm=l10d.lnflnonfarm
75 gen l11dlnflnonfarm=l11d.lnflnonfarm
76 gen l12dlnflnonfarm=l12d.lnflnonfarm
77 gen l24dlnflnonfarm=l24d.lnflnonfarm
78
79 gen dlnfl1f=d.lnfl1f
80 gen l1dlnfl1f=l1d.lnfl1f
81 gen l2dlnfl1f=l2d.lnfl1f
82 gen l3dlnfl1f=l3d.lnfl1f
83 gen l4dlnfl1f=l4d.lnfl1f
84 gen l5dlnfl1f=l5d.lnfl1f
85 gen l6dlnfl1f=l6d.lnfl1f
86 gen l7dlnfl1f=l7d.lnfl1f
87 gen l8dlnfl1f=l8d.lnfl1f
88 gen l9dlnfl1f=l9d.lnfl1f
89 gen l10dlnfl1f=l10d.lnfl1f
90 gen l11dlnfl1f=l11d.lnfl1f
91 gen l12dlnfl1f=l12d.lnfl1f
92 gen l24dlnfl1f=l24d.lnfl1f
93
94 gen dlnusepr=d.lnusepr
95 gen l1dlnusepr=l1d.lnusepr
96 gen l2dlnusepr=l2d.lnusepr
97 gen l3dlnusepr=l3d.lnusepr
```



```

98 gen l4dlnusepr=l4d.lnusepr
99 gen l5dlnusepr=l5d.lnusepr
100 gen l6dlnusepr=l6d.lnusepr
101 gen l7dlnusepr=l7d.lnusepr
102 gen l8dlnusepr=l8d.lnusepr
103 gen l9dlnusepr=l9d.lnusepr
104 gen l10dlnusepr=l10d.lnusepr
105 gen l11dlnusepr=l11d.lnusepr
106 gen l12dlnusepr=l12d.lnusepr
107 gen l24dlnusepr=l24d.lnusepr
108
109
110 gsreg dlnflnonfarm l1dlnflnonfarm l3dlnflnonfarm l6dlnflnonfarm l9dlnflnonfarm ///
111     l12dlnflnonfarm l24dlnflnonfarm ///
112     l1dlnfl1f l3dlnfl1f l6dlnfl1f l9dlnfl1f ///
113     l12dlnfl1f l24dlnfl1f ///
114     l1dlnusepr l3dlnusepr l6dlnusepr l9dlnusepr ///
115     l12dlnusepr l24dlnusepr if tin(1990m1,2019m12), ///
116     ncomb(1,6) aic outsample(24) fix(m1 m3 m6 m9) ///
117     samesample nindex( -1 aic -1 bic -1 rmse_out) results(gsreg_dlnrer) replace
118
119
120 *5
121 /*
122 Best model
123 reg dlnflnonfarm l3dlnflnonfarm l6dlnflnonfarm l12dlnflnonfarm l24dlnflnonfarm
124     l24dlnfl1f l6dlnusepr m1 m3 m6 m9
125 */
126 scalar drop _all
127 quietly forval w=48(12)144 {
128     gen pred=.
129     gen nobs=.
130     forval t=529/720 {
131         gen wstart=`t'-'w'
132         gen wend=`t'-1
133         reg d.lnflnonfarm l3d.lnflnonfarm l6d.lnflnonfarm l12d.lnflnonfarm
134             l24d.lnflnonfarm ///
135             l24d.lnfl1f l6d.lnusepr m1 m3 m6 m9 ///
136             if date>=wstart & date<=wend
137             replace nobs=e(N) if date==`t'
138             predict ptemp
139             replace pred=ptemp if date==`t'
140             drop ptemp wstart wend
141         }
142     gen errsq=(pred-d.lnflnonfarm)^2
143     summ errsq
144     scalar RWrmse`w'=r(mean)^.5
145     summ nobs
146     scalar RWminobs`w'=r(min)
147     scalar RWmaxobs`w'=r(max)
148     drop errsq pred nobs

```

```

148 }
149 scalar list
150 /*
151 RWmaxobs108 = 108
152 RWminobs108 = 108
153 RWrmse108 = .00388844
154 */
155
156 /*
157 Smallest / best model
158 reg dlnflnonfarm l12dlnflnonfarm m1 m3 m6 m9
159 */
160 scalar drop _all
161 quietly forval w=48(12)144 {
162   gen pred=.
163   gen nob=.
164   forval t=529/720 {
165     gen wstart=`t'-'w'
166     gen wend=`t'-1
167     reg dlnflnonfarm l12dlnflnonfarm m1 m3 m6 m9 ///
168     if date>=wstart & date<=wend
169     replace nob=e(N) if date==`t'
170     predict ptemp
171     replace pred=ptemp if date==`t'
172     drop ptemp wstart wend
173   }
174   gen errsq=(pred-d.lnflnonfarm)^2
175   summ errsq
176   scalar RWrmse`w'=r(mean)^.5
177   summ nob
178   scalar RWminobs`w'=r(min)
179   scalar RWmaxobs`w'=r(max)
180   drop errsq pred nob
181 }
182 scalar list
183 /*
184 RWmaxobs120 = 120
185 RWminobs120 = 120
186 RWrmse120 = .00423688
187
188 */
189
190 /*
191 Best medium length model
192 reg dlnflnonfarm l3dlnflnonfarm l12dlnflnonfarm l24dlnflnonfarm l6dlnusepr
193   m1 m3 m6 m9
194 */
195 scalar drop _all
196 quietly forval w=48(12)144 {
197   gen pred=.
198   gen nob=.

```

```

199   forval t=529/720 {
200     gen wstart=`t'-'`w'
201     gen wend=`t'-1
202     reg dlfnflnonfarm l3dlfnflnonfarm l12dlfnflnonfarm l24dlfnflnonfarm l6dlfnusepr ///
203       m1 m3 m6 m9 ///
204       if date>=wstart & date<=wend
205       replace nobse=e(N) if date==`t'
206       predict ptemp
207       replace pred=ptemp if date==`t'
208       drop ptemp wstart wend
209   }
210   gen errsq=(pred-d.lfnflnonfarm)^2
211   summ errsq
212   scalar RWrmse`w'=r(mean)^.5
213   summ nobse
214   scalar RWminobs`w'=r(min)
215   scalar RWmaxobs`w'=r(max)
216   drop errsq pred nobse
217 }
218 scalar list
219 /*
220 RWmaxobs108 = 108
221 RWminobs108 = 108
222 RWrmse108 = .00406403
223 */
224
225 *6
226 /*
227 RWmaxobs108 = 108
228 RWminobs108 = 108
229 RWrmse108 = .00388844
230 */
231 scalar drop _all
232 quietly forval w=156(12)156 {
233   gen pred=.
234   gen nobse=.
235   forval t=432/720 {
236     gen wstart=`t'-'`w'
237     gen wend=`t'-1
238     reg d.lfnflnonfarm l3d.lfnflnonfarm l6d.lfnflnonfarm l12d.lfnflnonfarm
239       l24d.lfnflnonfarm ///
240       l24d.lfnflf l6d.lfnusepr m1 m3 m6 m9 ///
241       if date>=wstart & date<=wend
242       replace nobse=e(N) if date==`t'
243       predict ptemp
244       replace pred=ptemp if date==`t'
245       drop ptemp wstart wend
246   }
247   gen errsq=(pred-d.lfnflnonfarm)^2
248   summ nobse // checking all had a full window

```

```

249 *get error info for normal interval
250 summ errsqr
251 scalar rwrmsqr=r(mean)^0.5
252 scalar list rwrmsqr
253 gen res=(d.lnflnonfarm-pred)
254 _pctile res, percentile(2.5,97.5)
255 return list
256
257 *7
258 predict temp if tin(2020m1,2020m1)
259 replace pred=temp if tin(2020m1,2020m1)
260 drop temp
261 gen pnonfarm=exp(1.lnflnonfarm+pred+(rwrmsqr^2)/2)
262 gen ubound=exp(1.lnflnonfarm+pred+1.96*rwrmsqr+(rwrmsqr^2)/2)
263 gen lbound=exp(1.lnflnonfarm+pred-1.96*rwrmsqr+(rwrmsqr^2)/2)
264 list month pnonfarm lbound ubound if tin(2020m1,2020m1)
265 tline pnonfarm lbound ubound fl_nonfarm if tin(2019m1,2020m1), tline(2019m12)
    saving("Nonfarm_Normal", replace)
266 graph export "nonfarm_normal.png", replace
267
268 *8
269 *Empirical
270 drop res
271 gen res=(d.lnflnonfarm-pred)
272 gen expres=exp(res)
273 summ expres
274 scalar meanexpres=r(mean)
275 gen pnonfarme=exp(1.lnflnonfarm+pred)*meanexpres
276 _pctile res, percentile(2.5,97.5)
277 return list
278 gen lboude=exp(1.lnflnonfarm+pred+r(r1))*meanexpres
279 gen uboude=exp(1.lnflnonfarm+pred+r(r2))*meanexpres
280 list month pnonfarme lboude uboude if tin(2020m1,2020m1)
281 tline pnonfarme lboude uboude fl_nonfarm if tin(2019m1,2020m1), ///
    tline(2019m12) saving("Nonfarm_Empirical", replace)
282 graph export "nonfarm_empirical.png", replace
283
284
285 *9
286 tline pnonfarm pnonfarme fl_nonfarm lbound lboude ubound uboude ///
    if tin(2019m1,2020m1), tline(2019m12) saving("Normal_vs_Empirical", replace)
287 graph export "normal_vs_empirical.png", replace
288
289
290 translate "Problem Set 5.smcl" "Problem Set 5.txt", replace
291 log close

```

Appendix B (STATA Output)

```
1 _____(R) _____
2 /_ / /_ / /_ /
3 ___/ / /___/ / /___/
4
5 Statistics/Data analysis
6 -----
7 -
8 name: <unnamed>
9 log: /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time
Series/Probl
10 > em Sets/Problem Set 5/Problem Set 5.smcl
11 log type: smcl
12 opened on: 3 Apr 2021, 21:50:15
13 1 . import delimited "Assignment_1_Monthly.txt"
14 (5 vars, 984 obs)
15
16 2 .
17 3 . rename lnu02300000 us_epr
18
19 4 . rename flnan fl_nonfarm
20
21 5 . rename fl1fn fl_lf
22
23 6 . rename flbppriv fl_bp
24
25 7 . rename date datestring
26
27 8 .
28 9 . gen datec=date(datestring, "YMD")
29
30 10 . gen date=mofd(datec)
31
32 11 . gen month=month(datec)
33
34 12 . format date %tm
35
36 13 .
37 14 . tsset date
38 time variable: date, 1939m1 to 2020m12
39 delta: 1 month
40
41 15 .
42 16 . gen lnusepr=log(us_epr)
```

```

43         (108 missing values generated)
44
45     17 . gen lnflnonfarm=log(fl_nonfarm)
46
47     18 . gen lnfllf=log(fl_lf)
48         (444 missing values generated)
49
50     19 . gen lnflbp=log(fl_bp)
51         (588 missing values generated)
52
53     20 .
54     21 . *1
55     22 . drop if !tin(1990m1,2019m12)
56         (624 observations deleted)
57
58     23 .
59     24 . *2
60     25 . tsset date
61             time variable:  date, 1990m1 to 2019m12
62                 delta:    1 month
63
64     26 . tsappend, add(1)
65
66     27 . replace month=month(dofm(date)) if month==.
67         (1 real change made)
68
69     28 .
70     29 . *interlude
71     30 . ac lnflnonfarm, saving("ac_lnflnonfarm.gph", replace)
72         (file ac_lnflnonfarm.gph saved)
73
74     31 . pac lnflnonfarm, saving("pac_lnflnonfarm.gph", replace)
75         (file pac_lnflnonfarm.gph saved)
76
77     32 . graph combine ac_lnflnonfarm.gph pac_lnflnonfarm.gph
78
79     33 . graph export ac_pac_lnflnonfarm.png, replace
80         (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
81         > /Problem Set 5/ac_pac_lnflnonfarm.png written in PNG format)
82
83     34 .
84     35 . *3
85     36 . gen m1=0
86
87     37 . replace m1=1 if month==1
88         (31 real changes made)
89
90     38 . gen m2=0
91
92     39 . replace m2=1 if month==2

```

```
93         (30 real changes made)
94
95     40 . gen m3=0
96
97     41 . replace m3=1 if month==3
98         (30 real changes made)
99
100    42 . gen m4=0
101
102    43 . replace m4=1 if month==4
103         (30 real changes made)
104
105    44 . gen m5=0
106
107    45 . replace m5=1 if month==5
108         (30 real changes made)
109
110    46 . gen m6=0
111
112    47 . replace m6=1 if month==6
113         (30 real changes made)
114
115    48 . gen m7=0
116
117    49 . replace m7=1 if month==7
118         (30 real changes made)
119
120    50 . gen m8=0
121
122    51 . replace m8=1 if month==8
123         (30 real changes made)
124
125    52 . gen m9=0
126
127    53 . replace m9=1 if month==9
128         (30 real changes made)
129
130    54 . gen m10=0
131
132    55 . replace m10=1 if month==10
133         (30 real changes made)
134
135    56 . gen m11=0
136
137    57 . replace m11=1 if month==11
138         (30 real changes made)
139
140    58 .
141    59 . gen dlnflnonfarm=d.lnflnonfarm
142         (2 missing values generated)
143
```

```
144 60 . gen l1dlnflnonfarm=l1d.lnflnonfarm
145     (2 missing values generated)
146
147 61 . gen l2dlnflnonfarm=l2d.lnflnonfarm
148     (3 missing values generated)
149
150 62 . gen l3dlnflnonfarm=l3d.lnflnonfarm
151     (4 missing values generated)
152
153 63 . gen l4dlnflnonfarm=l4d.lnflnonfarm
154     (5 missing values generated)
155
156 64 . gen l5dlnflnonfarm=l5d.lnflnonfarm
157     (6 missing values generated)
158
159 65 . gen l6dlnflnonfarm=l6d.lnflnonfarm
160     (7 missing values generated)
161
162 66 . gen l7dlnflnonfarm=l7d.lnflnonfarm
163     (8 missing values generated)
164
165 67 . gen l8dlnflnonfarm=l8d.lnflnonfarm
166     (9 missing values generated)
167
168 68 . gen l9dlnflnonfarm=l9d.lnflnonfarm
169     (10 missing values generated)
170
171 69 . gen l10dlnflnonfarm=l10d.lnflnonfarm
172     (11 missing values generated)
173
174 70 . gen l11dlnflnonfarm=l11d.lnflnonfarm
175     (12 missing values generated)
176
177 71 . gen l12dlnflnonfarm=l12d.lnflnonfarm
178     (13 missing values generated)
179
180 72 . gen l24dlnflnonfarm=l24d.lnflnonfarm
181     (25 missing values generated)
182
183 73 .
184 74 . gen dlnfl1f=d.lnfl1f
185     (2 missing values generated)
186
187 75 . gen l1dlnfl1f=l1d.lnfl1f
188     (2 missing values generated)
189
190 76 . gen l2dlnfl1f=l2d.lnfl1f
191     (3 missing values generated)
192
193 77 . gen l3dlnfl1f=l3d.lnfl1f
194     (4 missing values generated)
```



```
195
196 78 . gen l4dlnfllf=l4d.lnfllf
197     (5 missing values generated)
198
199 79 . gen l5dlnfllf=l5d.lnfllf
200     (6 missing values generated)
201
202 80 . gen l6dlnfllf=l6d.lnfllf
203     (7 missing values generated)
204
205 81 . gen l7dlnfllf=l7d.lnfllf
206     (8 missing values generated)
207
208 82 . gen l8dlnfllf=l8d.lnfllf
209     (9 missing values generated)
210
211 83 . gen l9dlnfllf=l9d.lnfllf
212     (10 missing values generated)
213
214 84 . gen l10dlnfllf=l10d.lnfllf
215     (11 missing values generated)
216
217 85 . gen l11dlnfllf=l11d.lnfllf
218     (12 missing values generated)
219
220 86 . gen l12dlnfllf=l12d.lnfllf
221     (13 missing values generated)
222
223 87 . gen l24dlnfllf=l24d.lnfllf
224     (25 missing values generated)
225
226 88 .
227 89 . gen dlnusepr=d.lnusepr
228     (2 missing values generated)
229
230 90 . gen l1dlnusepr=l1d.lnusepr
231     (2 missing values generated)
232
233 91 . gen l2dlnusepr=l2d.lnusepr
234     (3 missing values generated)
235
236 92 . gen l3dlnusepr=l3d.lnusepr
237     (4 missing values generated)
238
239 93 . gen l4dlnusepr=l4d.lnusepr
240     (5 missing values generated)
241
242 94 . gen l5dlnusepr=l5d.lnusepr
243     (6 missing values generated)
244
245 95 . gen l6dlnusepr=l6d.lnusepr
```

```

246         (7 missing values generated)
247
248     96 . gen l7dlgnosepr=l7d.lgnosepr
249         (8 missing values generated)
250
251     97 . gen l8dlgnosepr=l8d.lgnosepr
252         (9 missing values generated)
253
254     98 . gen l9dlgnosepr=l9d.lgnosepr
255         (10 missing values generated)
256
257     99 . gen l10dlgnosepr=l10d.lgnosepr
258         (11 missing values generated)
259
260    100 . gen l11dlgnosepr=l11d.lgnosepr
261         (12 missing values generated)
262
263    101 . gen l12dlgnosepr=l12d.lgnosepr
264         (13 missing values generated)
265
266    102 . gen l24dlgnosepr=l24d.lgnosepr
267         (25 missing values generated)
268
269    103 .
270    104 .
271    105 . gsreg dlgnflnonfarm l1dlgnflnonfarm l3dlgnflnonfarm l6dlgnflnonfarm
19dlgnflnonfarm
272         > m ///
273         >         l12dlgnflnonfarm l24dlgnflnonfarm ///
274         >         l1dlgnfl1f l3dlgnfl1f l6dlgnfl1f l9dlgnfl1f ///
275         >         l12dlgnfl1f l24dlgnfl1f ///
276         >         l1dlgnosepr l3dlgnosepr l6dlgnosepr l9dlgnosepr ///
277         >         l12dlgnosepr l24dlgnosepr if tin(1990m1,2019m12), ///
278         >         ncomb(1,6) aic outsample(24) fix(m1 m3 m6 m9) ///
279         >         samesample nindex( -1 aic -1 bic -1 rmse_out) results(gsreg_dlnrer)
r
280         > eplace
281         -----
282         Total Number of Estimations: 31179
283         -----
284         -----
285         Warning: Estimation could take about 14 minutes
286         -----
287
288         Computing combinations...
289         Preparing regression list...
290         Doing regressions...
291         Saving results...
292         file gsreg_dlnrer.dta saved
293         -----
294         Best estimation in terms of -1 aic -1 bic -1 rmse_out

```

```

295 Estimation number 19215
296 -----
297
298 Source | SS df MS Number of obs = 312
299 -----+----- F(10, 301) = 192.38
300 Model | .027480005 10 .002748 Prob > F = 0.0000
301 Residual | .004299465 301 .000014284 R-squared = 0.8647
302 -----+----- Adj R-squared = 0.8602
303 Total | .03177947 311 .000102185 Root MSE = .00378
304
305 -----
306 -
307 --
307 dlNFLnonfarm | Coef. Std. Err. t P>|t| [95% Conf.
Interva
308 > 1]
309 -----+-----
310 -
310 --
311 m1 | -.0068526 .0013736 -4.99 0.000 -.0095558
-.00414
312 > 94
313 m3 | .0009126 .0008591 1.06 0.289 -.0007781
.00260
314 > 33
315 m6 | -.0044233 .0010173 -4.35 0.000 -.0064252
-.00242
316 > 14
317 m9 | -.0008321 .0008569 -0.97 0.332 -.0025184
.00085
318 > 41
319 l3dlNFLnonfarm | .0992292 .0282426 3.51 0.001 .0436512
.15480
320 > 71
321 l6dlNFLnonfarm | .0741191 .0284776 2.60 0.010 .0180786
.13015
322 > 96
323 l12dlNFLnonfarm | .5446389 .067997 8.01 0.000 .4108292
.67844
324 > 87
325 l24dlNFLnonfarm | .1684682 .0598024 2.82 0.005 .0507844
.2861
326 > 52
327 l24dlNFLllf | -.1553647 .0508596 -3.05 0.002 -.2554501
-.05527
328 > 93
329 l6dlNusepr | .1231808 .0528785 2.33 0.020 .0191224
.22723
330 > 92
331 _cons | .0014172 .0003147 4.50 0.000 .0007979
.00203

```

```

332         > 65
333         -----
334         --
335
336     106 .
337     107 .
338     108 . *5
339     109 . /*
340         > Best model
341         > reg dlnflnonfarm l3dlnflnonfarm l6dlnflnonfarm l12dlnflnonfarm
124dlnflnonfar
342         > m
343         >          124dlnfl1lf l6dlnusepr m1 m3 m6 m9
344         > */
345     110 . scalar drop _all
346
347     111 . quietly forval w=48(12)144 {
348
349     112 . scalar list
350         RWmaxobs144 =          144
351         RWminobs144 =          144
352         RWrmse144 =   .00396645
353         RWmaxobs132 =          132
354         RWminobs132 =          132
355         RWrmse132 =   .00390407
356         RWmaxobs120 =          120
357         RWminobs120 =          120
358         RWrmse120 =   .00388926
359         RWmaxobs108 =          108
360         RWminobs108 =          108
361         RWrmse108 =   .00388844
362         RWmaxobs96 =           96
363         RWminobs96 =           96
364         RWrmse96 =   .00403691
365         RWmaxobs84 =           84
366         RWminobs84 =           84
367         RWrmse84 =   .00406426
368         RWmaxobs72 =           72
369         RWminobs72 =           72
370         RWrmse72 =   .00411873
371         RWmaxobs60 =           60
372         RWminobs60 =           60
373         RWrmse60 =   .00431692
374         RWmaxobs48 =           48
375         RWminobs48 =           48
376         RWrmse48 =   .00460352
377
378     113 . /*
379         > RWmaxobs108 = 108
380         > RWminobs108 = 108

```

```

381         > RWrmse108 = .00388844
382     > */
383 114 .
384 115 . /*
385     > Smallest / best model
386     > reg dlnflnonfarm l12dlnflnonfarm m1 m3 m6 m9
387     > */
388 116 . scalar drop _all
389
390 117 . quietly forval w=48(12)144 {
391
392 118 . scalar list
393     RWmaxobs144 =      144
394     RWminobs144 =      144
395     RWrmse144 = .00431666
396     RWmaxobs132 =      132
397     RWminobs132 =      132
398     RWrmse132 = .00426742
399     RWmaxobs120 =      120
400     RWminobs120 =      120
401     RWrmse120 = .00423688
402     RWmaxobs108 =      108
403     RWminobs108 =      108
404     RWrmse108 = .00428159
405     RWmaxobs96 =       96
406     RWminobs96 =       96
407     RWrmse96 = .00436091
408     RWmaxobs84 =       84
409     RWminobs84 =       84
410     RWrmse84 = .00439555
411     RWmaxobs72 =       72
412     RWminobs72 =       72
413     RWrmse72 = .00443487
414     RWmaxobs60 =       60
415     RWminobs60 =       60
416     RWrmse60 = .00453048
417     RWmaxobs48 =       48
418     RWminobs48 =       48
419     RWrmse48 = .00458215
420
421 119 . /*
422     > RWmaxobs120 = 120
423     > RWminobs120 = 120
424     > RWrmse120 = .00423688
425     >
426     > */
427 120 .
428 121 . /*
429     > Best medium length model
430     > reg dlnflnonfarm l3dlnflnonfarm l12dlnflnonfarm l24dlnflnonfarm l6dlnusepr
431     >           m1 m3 m6 m9

```

```

432     > */
433 122 . scalar drop _all
434
435 123 . quietly forval w=48(12)144 {
436
437 124 . scalar list
438     RWmaxobs144 =      144
439     RWminobs144 =      144
440     RWrmse144 = .00412303
441     RWmaxobs132 =      132
442     RWminobs132 =      132
443     RWrmse132 = .00407538
444     RWmaxobs120 =      120
445     RWminobs120 =      120
446     RWrmse120 = .00406735
447     RWmaxobs108 =      108
448     RWminobs108 =      108
449     RWrmse108 = .00406403
450     RWmaxobs96 =       96
451     RWminobs96 =       96
452     RWrmse96 = .00419684
453     RWmaxobs84 =       84
454     RWminobs84 =       84
455     RWrmse84 = .00423362
456     RWmaxobs72 =       72
457     RWminobs72 =       72
458     RWrmse72 = .00429113
459     RWmaxobs60 =       60
460     RWminobs60 =       60
461     RWrmse60 = .00448591
462     RWmaxobs48 =       48
463     RWminobs48 =       48
464     RWrmse48 = .00478837
465
466 125 . /*
467     > RWmaxobs108 = 108
468     > RWminobs108 = 108
469     > RWrmse108 = .00406403
470     > */
471 126 .
472 127 . *6
473 128 . /*
474     > RWmaxobs108 = 108
475     > RWminobs108 = 108
476     > RWrmse108 = .00388844
477     > */
478 129 . scalar drop _all
479
480 130 . quietly forval w=156(12)156 {
481
482 131 . summ nobobs // checking all had a full window

```

```

483
484      Variable |      Obs      Mean    Std. Dev.      Min      Max
485      -----+-----
486      nobs |      289    135.2561    32.98122      47     156
487
488 132 . *get error info for normal interval
489 133 . summ errsqr
490
491      Variable |      Obs      Mean    Std. Dev.      Min      Max
492      -----+-----
493      errsqr |      288      .000015    .0000471    1.10e-12    .0005431
494
495 134 . scalar rwrmsr=r(mean)^0.5
496
497 135 . scalar list rwrmsr
498      rwrmsr =    .00387308
499
500 136 . gen res=(d.lnflnonfarm-pred)
501      (73 missing values generated)
502
503 137 . _pctile res, percentile(2.5,97.5)
504
505 138 . return list
506
507      scalars:
508              r(r1) =   -.0074653569608927
509              r(r2) =    .0065394379198551
510
511 139 .
512 140 . *7
513 141 . predict temp if tin(2020m1,2020m1)
514      (option xb assumed; fitted values)
515      (360 missing values generated)
516
517 142 . replace pred=temp if tin(2020m1,2020m1)
518      (0 real changes made)
519
520 143 . drop temp
521
522 144 . gen pnonfarm=exp(l.lnflnonfarm+pred+(rwrmsr^2)/2)
523      (72 missing values generated)
524
525 145 . gen ubound=exp(l.lnflnonfarm+pred+1.96*rwrmsr+(rwrmsr^2)/2)
526      (72 missing values generated)
527
528 146 . gen lbound=exp(l.lnflnonfarm+pred-1.96*rwrmsr+(rwrmsr^2)/2)
529      (72 missing values generated)
530
531 147 . list month pnonfarm lbound ubound if tin(2020m1,2020m1)
532
533      +-----+

```

```

534         | month      pnonfarm      lbound      ubound |
535         |-----|
536     361. |      1    9001.077    8933.007    9069.667 |
537         +-----+
538
539     148 . tsline pnonfarm lbound ubound fl_nonfarm if tin(2019m1,2020m1),
tline(2019m12
540         > ) saving("Nonfarm_Normal", replace)
541         (file Nonfarm_Normal.gph saved)
542
543     149 . graph export "nonfarm_normal.png", replace
544         (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
545         > /Problem Set 5/nonfarm_normal.png written in PNG format)
546
547     150 .
548     151 . *8
549     152 . *Empirical
550     153 . drop res
551
552     154 . gen res=(d.lnflnonfarm-pred)
553         (73 missing values generated)
554
555     155 . gen expres=exp(res)
556         (73 missing values generated)
557
558     156 . summ expres
559
560         Variable |          Obs          Mean      Std. Dev.          Min          Max
561         -----+-----
562         expres |          288      .9997746      .0038735      .9777296      1.023579
563
564     157 . scalar meanexpres=r(mean)
565
566     158 . gen pnonfarme=exp(1.lnflnonfarm+pred)*meanexpres
567         (72 missing values generated)
568
569     159 . _pctile res, percentile(2.5,97.5)
570
571     160 . return list
572
573         scalars:
574                 r(r1) =  -.0074653569608927
575                 r(r2) =  .0065394379198551
576
577     161 . gen lboude=exp(1.lnflnonfarm+pred+r(r1))*meanexpres
578         (72 missing values generated)
579
580     162 . gen uboude=exp(1.lnflnonfarm+pred+r(r2))*meanexpres
581         (72 missing values generated)
582

```



```

583     163 . list month pnonfarme lbounde ubounde if tin(2020m1,2020m1)
584
585           +-----+
586           | month   pnonfa~e   lbounde   ubounde |
587           |-----|
588     361. |      1   8998.981   8932.051   9058.022 |
589           +-----+
590
591     164 . tsline pnonfarme lbounde ubounde fl_nonfarm if tin(2019m1,2020m1), ///
592           >         tline(2019m12) saving("Nonfarm_Epirical", replace)
593           (file Nonfarm_Epirical.gph saved)
594
595     165 . graph export "nonfarm_empirical.png", replace
596           (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
597           > /Problem Set 5/nonfarm_empirical.png written in PNG format)
598
599     166 .
600     167 . *9
601     168 . tsline pnonfarm pnonfarme fl_nonfarm lbound lbounde ubound ubounde ///
602           > if tin(2019m1,2020m1), tline(2019m12) saving("Normal_vs_Empirical",
replace)
603           (file Normal_vs_Empirical.gph saved)
604
605     169 . graph export "normal_vs_empirical.png", replace
606           (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
607           > /Problem Set 5/normal_vs_empirical.png written in PNG format)
608
609     170 .

```