

Final Project  
Gus Lipkin  
CAP 4763 Time Series Modelling and Forecasting  
22 April 2021

## Abstract

---

Time Series Modelling and Forecasting is a set of tools that allow us to use data from the past to create models of the past that can then be used to predict the future. In this paper, I use the data available for The Villages metropolitan statistical area (MSA) in Florida to try and predict the March 2021 numbers for the number of total employment for private employers and average weekly earnings. The data is gathered from the St Louis Federal Reserve Economic Data (FRED) website and analyzed with a series of Time Series tools in STATA.

# Table of Contents

---

Section
<a href="#">Abstract</a>
<a href="#">Table of Contents</a>
<a href="#">Introduction</a>
<a href="#">Data</a>
<a href="#">Model Estimation and Selection</a>
<a href="#">Final Results</a>
<a href="#">Conclusion</a>
<a href="#">Appendix A</a>
<a href="#">Appendix B</a>

# Introduction

---

In CAP 4763 Time Series Modelling and Forecasting we learned about what it takes to create a time series model and forecast one or more periods into the future using the models we've made. Our final project is the culmination of all we've learned in the class and the goal of our project is to predict two values for a single metropolitan statistical area (MSA) in Florida. I chose The Villages because of its status as the largest retirement community in the country. I thought it would be interesting to study the trends for total private employment and average weekly earnings for a community that is composed primarily of retirees. Because I knew nothing else about The Villages, I had no preconceived notions about what the data might show me or how I could best use the data available to estimate the variables in question.

Our task was to estimate the values for March 2021 which the St Louis FRED would release the real values for shortly after the paper is due. This would allow us to test our skills against real world data. FRED was a bit excited about the March data and released it ten days early which allows me to compare my forecasts against the real world data. Throughout the paper I'll discuss how this was made possible with summary statistics, auto and partial autocorrelograms, time series plots, global search regression, rolling window forecasts, and fan charts.

## Data

---

To start, I downloaded all available data for private, non-seasonally adjusted, monthly data for all time for The Villages MSA from FRED. Data for private service employees and all private employees was available going all the way back to January 1990 and all the way through March 2021. Average hourly and weekly earnings and weekly average hours for all private employees is available from January 2011 to March 2021. When the project started, February and March 2021 data was not available yet. Once they became available, I re-downloaded the data in TSV format from FRED.

To work with the data I first had to change the variables to something more usable with the `rename` command in STATA. Next, we had to change the data into something that is recognized as time series data. I used `generate` to create the appropriate monthly date variables and `tsset` the data so that it was recognized as starting at January 2021, the earliest datapoint in the data. Because I am predicting March 2021 and can't predict it with the data, I created a set of dummy columns for total private employment and weekly earnings that had the March data and dropped the March data for the main data columns. This will allow me to compare my forecast to the actual value.

Performing log transforms on all the variables was the next data manipulation step. Log transforms serve two purposes. The first is to normalize the data so that it forms a bell curve. The second is to force all models and forecasts to only produce positive values because unless something has gone very very wrong, employment and weekly earnings will both always be positive.

The number of total employment and service employment are both in the thousands of numbers while the number of hours is in hours and the number of earnings is in dollars.

## Summary Statistics

---

Calculating summary statistics for all variables can provide some valuable base insights that I can use to help shape my models.

## Table 1 Summary Statistics for All Standard Variables

Variable	Obs	Mean	Std. Dev.	Min	Max
Count	374	14.18556	6.880684	5.3	28
WeekHours	123	36.88455	3.791817	28.3	45.8
HourlyEarnings	123	19.72	2.903968	15.01	24.6
WeeklyEarnings	122	719.6542	84.57241	503.79	916.1
ServiceCount	375	10.43387	5.959179	3.9	22.8

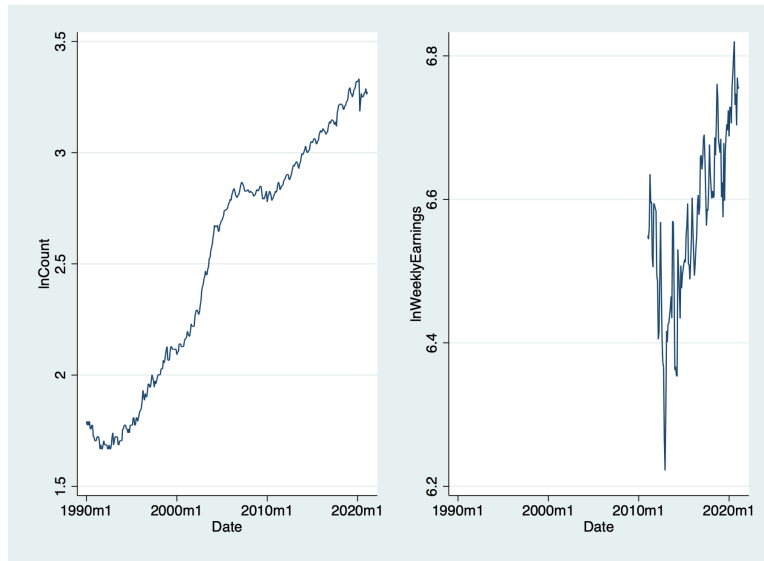
Of particular note in the non-transformed data is that there is a wide variation in the number of private employment for The Villages has an incredibly high range where the maximum value is more than five times greater than the minimum value. Meanwhile, hourly earnings have not increased by nearly the same factor. This indicates that The Villages are growing and supply is increasing to meet rising demand from continued development in The Villages MSA.

## Table 2 Summary Statistics for All Log Transformed Variables

Variable	Obs	Mean	Std. Dev.	Min	Max
InCount	374	2.5174	.5398403	1.667707	3.332205
InWeekHours	123	3.602488	.10385	3.342862	3.824284
InHourlyEarnings	123	2.970779	.1482819	2.708717	3.202746
InWeeklyEarnings	122	6.571775	.1195694	6.222159	6.820126
InServiceCount	375	2.172053	.5985689	1.360977	3.12676

To a more trained eye, the log transformed data may provide more insights, but I am not that good.

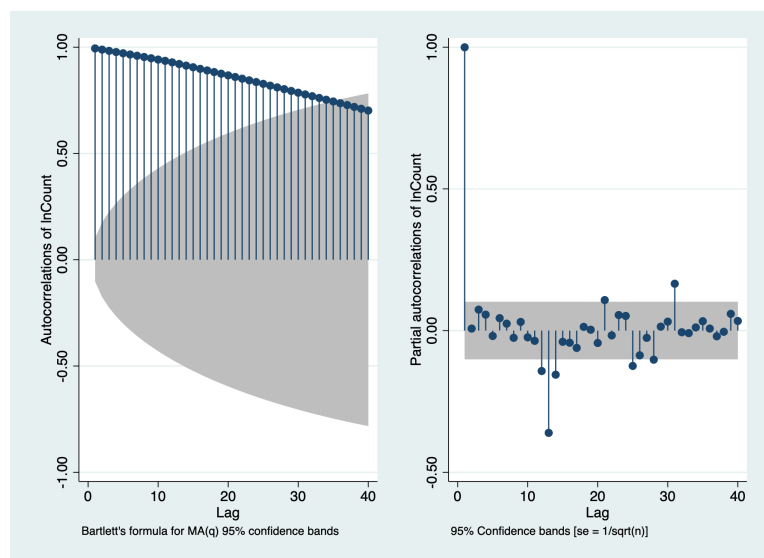
## Figure 1 Time Series Plots of InCount and InWeeklyEarnings



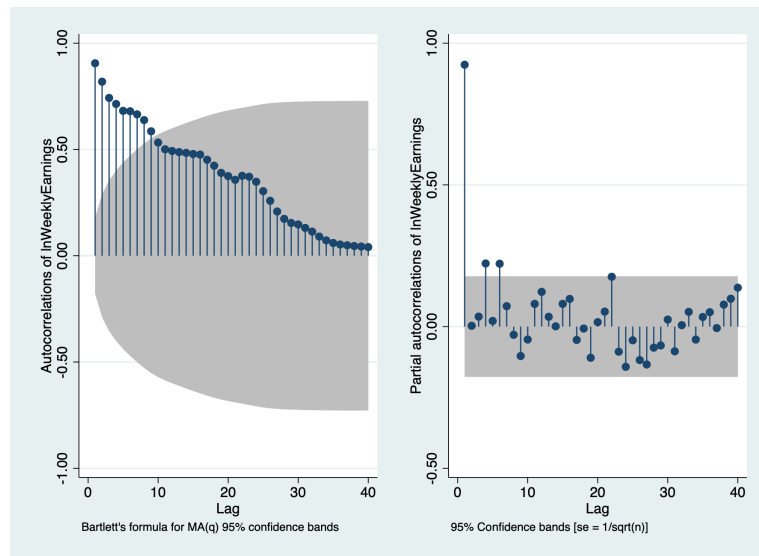
The first step after creating summary statistics is to generate a time series line plot so we can see how the data is changing over time and give context to the summary statistics. On the left, we see the log transform of the number of private employment in The Villages. There is a solid upwards trend except for the market crashes in the early 1990s, late 2000s, and in 2020. Looking at `lnWeeklyEarnings` on the right, the data does not start until 2011 but the graph has been adjusted to show the same time period as `lnCount`. The data sharply decreases around 2012 and then slowly rises on average with large differences in month to month data. This could be due to The Villages being populated by old people who travel to Florida from their home state for the winter months, also known as "snow-birding."

Next, we want to make sure the data is stationary. Stationary data has a constant mean, variance, and autocorrelation through time which makes the data easier to work with, especially in time series modelling and forecasting. To solve this, we difference the data. Differencing is when you find the difference between each datapoint and the previous one. We can check to see if we need to difference our data by graphing the autocorrelogram and partial autocorrelogram. Below are the AC and PAC for `lnCount` and `lnWeekly Earnings`.

**Figure 2 AC and PAC of `lnCount`**



### Figure 3 AC and PAC of lnWeeklyEarnings



Both AC charts show a high first term and then steadily decreasing terms thereafter which suggests that there is an autoregressive term in the data. Autoregressive data is data that is dependent, at least partially, on the data before it. Recursive formulas in geometric sequences are a good example. The PAC shows a significant value and then alternating positive and negative insignificant values, characteristic of a higher order moving average term. That is to say, the data has trends that repeat. In this case, it is likely seasonal trends such as increased employees during the holidays.

## Model Estimation and Selection

Model estimation and selection is the keystone in time series modeling and forecasting. No matter how well prepared you are, if you cannot construct a good model, then all of your labor will be for nothing. Using a combination of intuition, global search regression, and rolling window forecasts, I was able to choose between a few different models that I thought would accurately model private employment and weekly earnings.

## Model Estimation and Selection Total Private Employment

Before diving in with more advanced techniques, I devised a few models that I thought might work well based on the time series plots, AC, and PAC that I generated earlier. My first model regressed the differenced value of `lnCount` against the twelfth, twenty-fourth, thirty-sixth, and forty-eighth lags of the differenced `lnCount`. The lags mean that the independent variables are `lnCount` from those many months ago each to create an autoregressive model. The final model used in STATA was `reg d.lnCount 1(12,24,36,48)d.lnCount`. When run through a rolling window forecast to find the optimal window size, I found that this initial model had an optimal window size of 132 months which resulted in a rolling window root mean square error of 0.0172128.

My next intuitive model for `lnCount` uses the fifth, twelfth, twenty-fourth, thirty-sixth, and forty-eighth lags of differenced `lnCount` and the fifth lag of different average weekly hours along with an indicator variable for May. The final regression in stata was `reg d.lnCount 1(5,12,24,36,48)d.lnCount 1(5)d.lnWeekHours m5`. With an optimal window size of eighty-four periods, I had a rolling window root mean square error of 0.01950911.

Begrudgingly, I moved on from my intuition to global search regression which allows me to input many different variables and run regressions on all combinations of the given variables. Again, I used the differenced `lnCount` as the dependent variable and the first twelve lags of the differenced `lnCount` along with the differenced twenty-fourth, thirty-sixth, and forty-eighth lags of `lnCount`. I chose to only use lagged versions of `lnCount` because service employment is just a subset of the total employment. The other variables I chose not to include because I wanted to make sure the search did not take too long to run and because the AC and PAC indicated high amounts of autoregression. I also fixed all month indicators.

**Table 3 GSREG Models and Results for Total Private Employment**

Model	Windows	RWRMSE
<code>reg d.lnCount l12d.lnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11</code>	144	0.01824906
<code>reg d.lnCount l(12,36)d.lnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11</code>	144	0.01777071
<code>reg d.lnCount l4d.lnWeekHours l9d.lnWeekHours l8d.lnHourlyEarnings m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11</code>	12	0.0176238

## Model Estimation and Selection Average Weekly Earnings

When modeling average weekly earnings, I used similar methods to total private employment. In a change of pace, however, I only guesstimated one model. I regressed the differenced `lnWeeklyEarnings` against the lagged and differenced `lnWeekHours` and `lnHourlyEarnings`. I was hoping that the combination of hours per week and earnings per hour would accurately predict the current weekly earnings. This could have worked better if I had weekly data but I do the best that I can. After I can the rolling window estimate, I found that the optimal window size was sixty months with a RWRMSE of 0.06145693.

For my global search regression, I used the same lags as with `lnCount` but instead did not use the forty-eighth lag for `lnWeeklyEarnings`. Even though I had the opportunity to use `lnWeekHours` and `lnHourlyEarnings` together as I suggested earlier, I chose to work with an autoregressive model again because as I also said before, the data is not weekly which makes those numbers a bit less valuable.

**Table 4 GSREG Models and Results for Average Weekly Earnings**

Model	Windows	RWRMSE
<code>reg d.lnWeeklyEarnings l3d.lnWeeklyEarnings l15d.lnWeeklyEarnings m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11</code>	84	0.06004448
<code>reg d.lnWeeklyEarnings l3d.lnWeeklyEarnings l15d.lnWeeklyEarnings l17d.lnWeeklyEarnings</code>	84	0.05250414

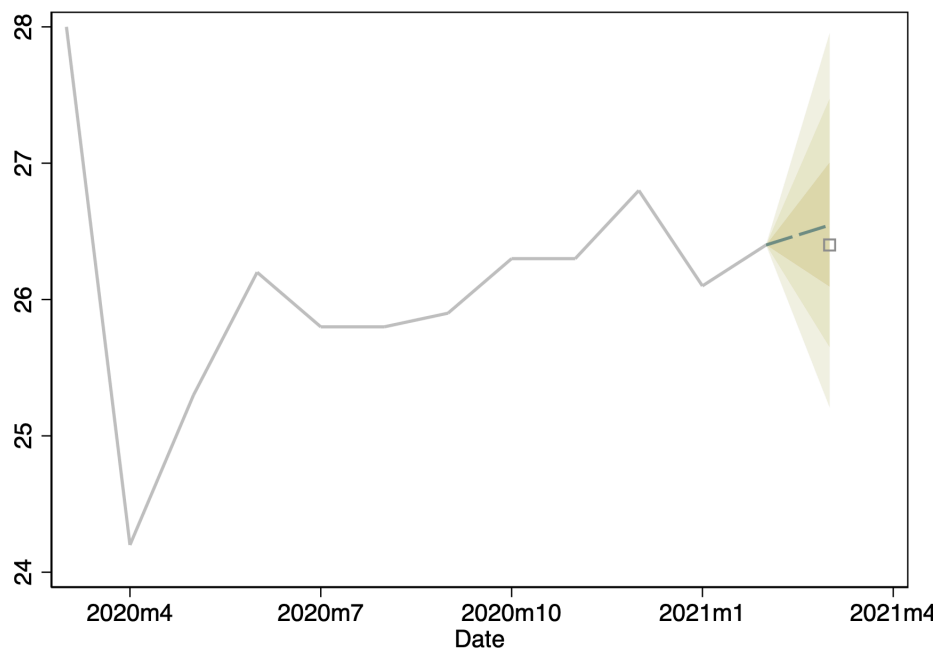
## Final Results

This year is especially exciting to be a student of the magnificent Dr Dewey because I can now compare my models to the real value. When selecting the models, I did not use the March data available to me because it would cloud the models and of course I can predict March's values if I know what March is.

## Final Results Total Private Employment

FRED says that the total private employment for March 2021 is 26,400. My best model `reg d.lnCount 1(12,24,36,48)d.lnCount`, predicted that the value would be 26,545 which is .5% higher than the real value. With real world context added, it is possible that the number is lower than I expected because many people received a third COVID stimulus check in March and some people used that opportunity to leave their jobs and begin a search for a new job that is better for them.

### Figure 4 Fan Chart of Total Private Employment

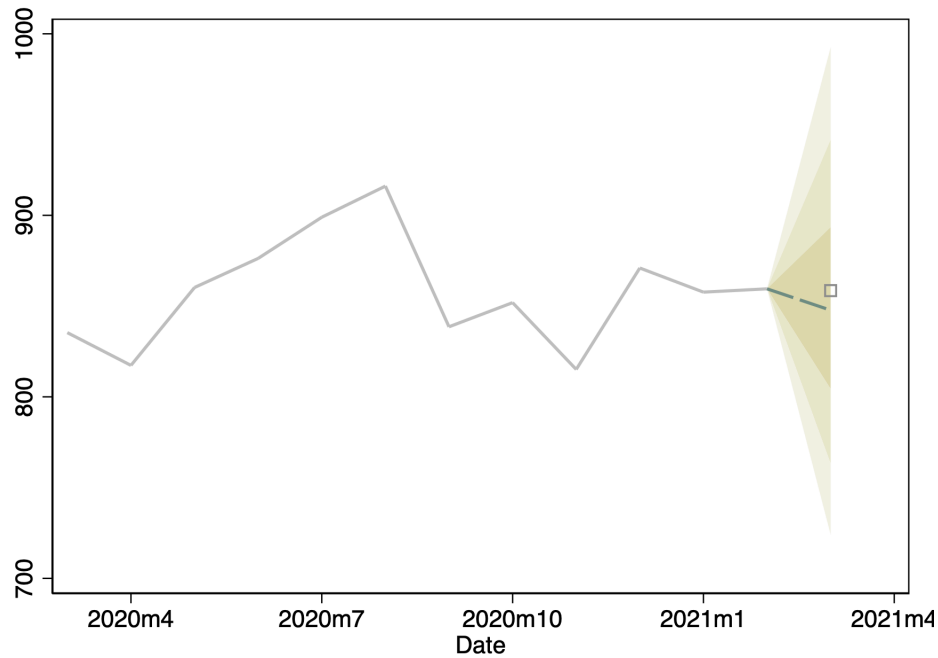


## Final Results Average Weekly Earnings

For average weekly earnings, FRED reported that in March 2021, the mean weekly income in The Villages was \$858.52. My estimate with `reg d.lnWeeklyEarnings 1(3,5,7)d.lnWeeklyEarnings` was \$847.7501 which is a difference of 1.25%. I believe my autoregressive model was a poor choice because here it is clearly influenced by a dip in weekly earnings in the last six months.

### Figure 5 Fan Chart of Average Weekly Earnings





## Conclusion

Time series modelling and forecasting is a complicated art but not nearly as complicated as it may seem initially. After a brief examination of the data with summary statistics, I examined the data visually with time series plots, autocorrelograms, and partial autocorrelograms. The ACs and PACs indicated that the data was not stationary and needed to be differenced. Next, using a mix of guesstimation and global search regression, I identified several models for each variable that I wanted to forecast. I ran each model through a rolling window forecast to identify the optimal window width and then used that to forecast the values for March 2021. While my forecasts weren't perfect, they were pretty good and well within the 95% confidence interval of the fan chart. If I were to improve the models for the future, I would introduce new variables to the models so that they weren't purely autoregressive.

## Appendix A

```
1 clear
2 set more off
3
4 cd "/Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem Sets/Final
  Project"
5 log using "Final Project.smcl", replace
6 import delimited "TS2020_Final_Project_Monthly.txt"
7 rename smu12455400500000001 Count
8 rename smu12455400500000002 WeekHours
9 rename smu12455400500000003 HourlyEarnings
10 rename smu12455400500000011 WeeklyEarnings
11 rename smu12455400800000001 ServiceCount
12
13
14 label variable Count "Count"
```

```

15 label variable WeekHours "WeekHours"
16 label variable HourlyEarnings "HourlyEarnings"
17 label variable WeeklyEarnings "WeeklyEarnings"
18 label variable ServiceCount "ServiceCount"
19
20
21 gen datec=date(date, "YMD")
22 gen Date=mofd(datec)
23 gen month=month(datec)
24 format Date %tm
25 tsset Date
26
27 gen withMarchCount = Count
28 gen withMarchEarnings = WeeklyEarnings
29 replace Count=. if tin(2021m3,)
30 replace WeeklyEarnings=. if tin(2021m3,)
31
32 gen lnCount = ln(Count)
33 gen lnWeekHours = ln(WeekHours)
34 gen lnHourlyEarnings = ln(HourlyEarnings)
35 gen lnWeeklyEarnings = ln(WeeklyEarnings)
36 gen lnServiceCount = ln(ServiceCount)
37
38 gen m1=0
39 replace m1=1 if month==1
40 gen m2=0
41 replace m2=1 if month==2
42 gen m3=0
43 replace m3=1 if month==3
44 gen m4=0
45 replace m4=1 if month==4
46 gen m5=0
47 replace m5=1 if month==5
48 gen m6=0
49 replace m6=1 if month==6
50 gen m7=0
51 replace m7=1 if month==7
52 gen m8=0
53 replace m8=1 if month==8
54 gen m9=0
55 replace m9=1 if month==9
56 gen m10=0
57 replace m10=1 if month==10
58 gen m11=0
59 replace m11=1 if month==11
60 gen m12=0
61 replace m12=1 if month==12
62
63 gen dlnCount=d.lnCount
64 gen l1dlnCount=l1d.lnCount
65 gen l2dlnCount=l2d.lnCount

```

```
66 gen l3dlnCount=l3d.lnCount
67 gen l4dlnCount=l4d.lnCount
68 gen l5dlnCount=l5d.lnCount
69 gen l6dlnCount=l6d.lnCount
70 gen l7dlnCount=l7d.lnCount
71 gen l8dlnCount=l8d.lnCount
72 gen l9dlnCount=l9d.lnCount
73 gen l10dlnCount=l10d.lnCount
74 gen l11dlnCount=l11d.lnCount
75 gen l12dlnCount=l12d.lnCount
76 gen l24dlnCount=l24d.lnCount
77 gen l36dlnCount=l36d.lnCount
78 gen l48dlnCount=l48d.lnCount
79
80 gen dlnWeekHours=d.lnWeekHours
81 gen l1dlnWeekHours=l1d.lnWeekHours
82 gen l2dlnWeekHours=l2d.lnWeekHours
83 gen l3dlnWeekHours=l3d.lnWeekHours
84 gen l4dlnWeekHours=l4d.lnWeekHours
85 gen l5dlnWeekHours=l5d.lnWeekHours
86 gen l6dlnWeekHours=l6d.lnWeekHours
87 gen l7dlnWeekHours=l7d.lnWeekHours
88 gen l8dlnWeekHours=l8d.lnWeekHours
89 gen l9dlnWeekHours=l9d.lnWeekHours
90 gen l10dlnWeekHours=l10d.lnWeekHours
91 gen l11dlnWeekHours=l11d.lnWeekHours
92 gen l12dlnWeekHours=l12d.lnWeekHours
93 gen l24dlnWeekHours=l24d.lnWeekHours
94 gen l36dlnWeekHours=l36d.lnWeekHours
95 gen l48dlnWeekHours=l48d.lnWeekHours
96
97 gen dlnHourlyEarnings=d.lnHourlyEarnings
98 gen l1dlnHourlyEarnings=l1d.lnHourlyEarnings
99 gen l2dlnHourlyEarnings=l2d.lnHourlyEarnings
100 gen l3dlnHourlyEarnings=l3d.lnHourlyEarnings
101 gen l4dlnHourlyEarnings=l4d.lnHourlyEarnings
102 gen l5dlnHourlyEarnings=l5d.lnHourlyEarnings
103 gen l6dlnHourlyEarnings=l6d.lnHourlyEarnings
104 gen l7dlnHourlyEarnings=l7d.lnHourlyEarnings
105 gen l8dlnHourlyEarnings=l8d.lnHourlyEarnings
106 gen l9dlnHourlyEarnings=l9d.lnHourlyEarnings
107 gen l10dlnHourlyEarnings=l10d.lnHourlyEarnings
108 gen l11dlnHourlyEarnings=l11d.lnHourlyEarnings
109 gen l12dlnHourlyEarnings=l12d.lnHourlyEarnings
110 gen l24dlnHourlyEarnings=l24d.lnHourlyEarnings
111 gen l36dlnHourlyEarnings=l36d.lnHourlyEarnings
112 gen l48dlnHourlyEarnings=l48d.lnHourlyEarnings
113
114 gen dlnWeeklyEarnings=d.lnWeeklyEarnings
115 gen l1dlnWeeklyEarnings=l1d.lnWeeklyEarnings
116 gen l2dlnWeeklyEarnings=l2d.lnWeeklyEarnings
```

```

117 gen l3dlnWeeklyEarnings=l3d.lnWeeklyEarnings
118 gen l4dlnWeeklyEarnings=l4d.lnWeeklyEarnings
119 gen l5dlnWeeklyEarnings=l5d.lnWeeklyEarnings
120 gen l6dlnWeeklyEarnings=l6d.lnWeeklyEarnings
121 gen l7dlnWeeklyEarnings=l7d.lnWeeklyEarnings
122 gen l8dlnWeeklyEarnings=l8d.lnWeeklyEarnings
123 gen l9dlnWeeklyEarnings=l9d.lnWeeklyEarnings
124 gen l10dlnWeeklyEarnings=l10d.lnWeeklyEarnings
125 gen l11dlnWeeklyEarnings=l11d.lnWeeklyEarnings
126 gen l12dlnWeeklyEarnings=l12d.lnWeeklyEarnings
127 gen l24dlnWeeklyEarnings=l24d.lnWeeklyEarnings
128 gen l36dlnWeeklyEarnings=l36d.lnWeeklyEarnings
129 gen l48dlnWeeklyEarnings=l48d.lnWeeklyEarnings
130
131 gen dlnServiceCount=d.lnServiceCount
132 gen l1dlnServiceCount=l1d.lnServiceCount
133 gen l2dlnServiceCount=l2d.lnServiceCount
134 gen l3dlnServiceCount=l3d.lnServiceCount
135 gen l4dlnServiceCount=l4d.lnServiceCount
136 gen l5dlnServiceCount=l5d.lnServiceCount
137 gen l6dlnServiceCount=l6d.lnServiceCount
138 gen l7dlnServiceCount=l7d.lnServiceCount
139 gen l8dlnServiceCount=l8d.lnServiceCount
140 gen l9dlnServiceCount=l9d.lnServiceCount
141 gen l10dlnServiceCount=l10d.lnServiceCount
142 gen l11dlnServiceCount=l11d.lnServiceCount
143 gen l12dlnServiceCount=l12d.lnServiceCount
144 gen l24dlnServiceCount=l24d.lnServiceCount
145 gen l36dlnServiceCount=l36d.lnServiceCount
146 gen l48dlnServiceCount=l48d.lnServiceCount
147
148 /*
149 The project is to forecast the March non-seasonally adjusted estimates of average
    weekly earnings and total employment for private employers (total private) for a
    Florida MSA of your choice and write up a professional report on your forecast.
150 */
151 /* Count and WeeklyEarnings */
152
153 summ Count WeekHours HourlyEarnings WeeklyEarnings ServiceCount
154 summ lnCount lnWeekHours lnHourlyEarnings lnWeeklyEarnings lnServiceCount
155
156 ac lnCount, saving(lnCount_ac, replace)
157 pac lnCount, saving(lnCount_pac, replace)
158 graph combine lnCount_ac.gph lnCount_pac.gph, saving(lnCount_ac_pac, replace)
159 graph export "lnCount_ac_pac.png", replace
160 ** Probably need to difference
161
162 ac lnWeeklyEarnings, saving(lnWeeklyEarnings_ac, replace)
163 pac lnWeeklyEarnings, saving(lnWeeklyEarnings_pac, replace)
164 graph combine lnWeeklyEarnings_ac.gph lnWeeklyEarnings_pac.gph,
    saving(lnWeeklyEarnings_ac_pac, replace)

```

```

165 graph export "lnWeeklyEarnings_ac_pac.png", replace
166 ** Probably need to difference
167
168 *starter models for count
169 *I used a pair plot to examine the rise and fall of variables with respect to each
    other
170 reg d.lnCount l(12,24,36,48)d.lnCount // .01637
171 scalar drop _all
172 quietly forval w=12(12)144 {
173     gen pred=.
174     gen nob=.
175     forval t=421/733 {
176         gen wstart=`t'-'w'
177         gen wend=`t'-1
178         reg dlnCount l12dlnCount l24dlnCount l36dlnCount l48dlnCount ///
179             if Date>=wstart & Date<=wend
180         replace nob=e(N) if Date==`t'
181         predict ptemp
182         replace pred=ptemp if Date==`t'
183         drop ptemp wstart wend
184     }
185     gen errsq=(pred-d.lnCount)^2
186     summ errsq
187     scalar RWrmse`w'=r(mean)^.5
188     summ nob
189     scalar RWminobs`w'=r(min)
190     scalar RWmaxobs`w'=r(max)
191     drop errsq pred nob
192 }
193 scalar list
194 /*
195 RWmaxobs132 =          132
196 RWminobs132 =          12
197 RWrmse132 =    .0172128
198 */
199
200 reg d.lnCount l(5,12,24,36,48)d.lnCount l(5)d.lnWeekHours m5 // .01711
201 scalar drop _all
202 quietly forval w=12(12)84 {
203     gen pred=.
204     gen nob=.
205     forval t=641/733 {
206         gen wstart=`t'-'w'
207         gen wend=`t'-1
208         reg dlnCount l5dlnCount l12dlnCount l24dlnCount l36dlnCount l48dlnCount
15dlnWeekHours m5 ///
209             if Date>=wstart & Date<=wend
210         replace nob=e(N) if Date==`t'
211         predict ptemp
212         replace pred=ptemp if Date==`t'
213         drop ptemp wstart wend

```

```

214     }
215     gen errsq=(pred-d.lnCount)^2
216     summ errsq
217     scalar RWrmse`w'=r(mean)^.5
218     summ nob
219     scalar RWminobs`w'=r(min)
220     scalar RWmaxobs`w'=r(max)
221     drop errsq pred nob
222 }
223 scalar list
224 /*
225 RWmaxobs84 =          84
226 RWminobs84 =          23
227 RWrmse84 =   .01950911
228 */
229
230 quietly gsreg dlnCount l1dlnCount l2dlnCount l3dlnCount l4dlnCount l5dlnCount
l6dlnCount ///
231     l7dlnCount l8dlnCount l9dlnCount l10dlnCount l11dlnCount l12dlnCount ///
232     l24dlnCount l36dlnCount l48dlnCount ///
233     if tin(1990m1,2021m1), ///
234     ncomb(1,12) aic outsample(24) fix(m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 m12) ///
235     samesample nindex( -1 aic -1 bic -1 rmse_out) results(gsreg_dlnCount) replace
236
237 *gsreg suggestions
238 reg d.lnCount l12d.lnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11
239 scalar drop _all
240 quietly forval w=12(12)144 {
241     gen pred=.
242     gen nob=.
243     forval t=385/733 {
244         gen wstart=`t'-'w'
245         gen wend=`t'-1
246         reg dlnCount l12dlnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 ///
247             if Date>=wstart & Date<=wend
248         replace nob=e(N) if Date==`t'
249         predict ptemp
250         replace pred=ptemp if Date==`t'
251         drop ptemp wstart wend
252     }
253     gen errsq=(pred-d.lnCount)^2
254     summ errsq
255     scalar RWrmse`w'=r(mean)^.5
256     summ nob
257     scalar RWminobs`w'=r(min)
258     scalar RWmaxobs`w'=r(max)
259     drop errsq pred nob
260 }
261 scalar list
262 /*
263 RWmaxobs144 =          144

```

```

264 RWminobs144 =          12
265 RWrmse144 =   .01824906
266 */
267
268 reg d.lnCount l(12,36)d.lnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11
269 scalar drop _all
270 quietly forval w=12(12)144 {
271   gen pred=.
272   gen nob=.
273   forval t=409/733 {
274     gen wstart=`t'-'w'
275     gen wend=`t'-1
276     reg dlnCount l12dlnCount l36dlnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11 ///
277     if Date>=wstart & Date<=wend
278     replace nob=e(N) if Date==`t'
279     predict ptemp
280     replace pred=ptemp if Date==`t'
281     drop ptemp wstart wend
282   }
283   gen errsq=(pred-d.lnCount)^2
284   summ errsq
285   scalar RWrmse`w'=r(mean)^.5
286   summ nob
287   scalar RWminobs`w'=r(min)
288   scalar RWmaxobs`w'=r(max)
289   drop errsq pred nob
290 }
291 scalar list
292 /*
293 RWmaxobs144 =          144
294 RWminobs144 =          12
295 RWrmse144 =   .01777071
296 */
297
298 reg d.lnCount l4d.lnWeekHours l9d.lnWeekHours l8d.lnHourlyEarnings m1 m2 m3 m4 m5 m6
m7 m8 m9 m10 m11
299 scalar drop _all
300 quietly forval w=12(12)84 {
301   gen pred=.
302   gen nob=.
303   forval t=624/733 {
304     gen wstart=`t'-'w'
305     gen wend=`t'-1
306     reg dlnCount l4dlnWeekHours l9dlnWeekHours l8dlnHourlyEarnings m1 m2 m3 m4 m5 m6
m7 m8 m9 m10 m11 ///
307     if Date>=wstart & Date<=wend
308     replace nob=e(N) if Date==`t'
309     predict ptemp
310     replace pred=ptemp if Date==`t'
311     drop ptemp wstart wend
312   }

```

```

313 gen errsq=(pred-d.lnCount)^2
314 summ errsq
315 scalar RWrmse`w'=r(mean)^.5
316 summ nob
317 scalar RWminobs`w'=r(min)
318 scalar RWmaxobs`w'=r(max)
319 drop errsq pred nob
320 }
321 scalar list
322 /*
323 RWmaxobs12 =          12
324 RWminobs12 =           2
325 RWrmse12 =    .0176238
326 */
327
328 scalar rwrms = .0172128
329 reg d.lnCount l(12,24,36,48)d.lnCount if tin(,2021m2)
330 predict pd
331 gen pflcount=exp((rwrms^2)/2)*exp(1.lnCount+pd) if Date==tm(2021m3)
332 gen ub1=exp((rwrms^2)/2)*exp(1.lnCount+pd+1*rwrms) if Date==tm(2021m3)
333 gen lb1=exp((rwrms^2)/2)*exp(1.lnCount+pd-1*rwrms) if Date==tm(2021m3)
334 gen ub2=exp((rwrms^2)/2)*exp(1.lnCount+pd+2*rwrms) if Date==tm(2021m3)
335 gen lb2=exp((rwrms^2)/2)*exp(1.lnCount+pd-2*rwrms) if Date==tm(2021m3)
336 gen ub3=exp((rwrms^2)/2)*exp(1.lnCount+pd+3*rwrms) if Date==tm(2021m3)
337 gen lb3=exp((rwrms^2)/2)*exp(1.lnCount+pd-3*rwrms) if Date==tm(2021m3)
338 drop pd
339
340 replace pflcount=Count if Date==tm(2021m2)
341 replace ub1=Count if Date==tm(2021m2)
342 replace ub2=Count if Date==tm(2021m2)
343 replace ub3=Count if Date==tm(2021m2)
344 replace lb1=Count if Date==tm(2021m2)
345 replace lb2=Count if Date==tm(2021m2)
346 replace lb3=Count if Date==tm(2021m2)
347
348 twoway (tsrline ub3 ub2 if tin(2020m3,2021m3), ///
349   recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
350   (tsrline ub2 ub1 if tin(2020m3,2021m3), ///
351   recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
352   (tsrline ub1 pflcount if tin(2020m3,2021m3), ///
353   recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///
354   (tsrline pflcount lb1 if tin(2020m3,2021m3), ///
355   recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///
356   (tsrline lb1 lb2 if tin(2020m3,2021m3), ///
357   recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
358   (tsrline lb2 lb3 if tin(2020m3,2021m3), ///
359   recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
360   (tsline Count pflcount if tin(2020m3,2021m3) , ///
361   lcolor(gs12 teal) lwidth(medthick medthick) ///
362   lpattern(solid longdash)) ///
363   (scatter withMarchCount Date if tin(2021m3,)), scheme(slmono) legend(off)

```



```

364 graph export "CountFan.png", replace
365
366 /*-----*/
367
368 *starter models for weekly earnings
369 reg d.lnWeeklyEarnings l1d.lnWeekHours ld.lnHourlyEarnings
370 scalar drop _all
371 quietly forval w=12(12)84 {
372   gen pred=.
373   gen nob=.
374   forval t=616/733 {
375     gen wstart=`t'-'w'
376     gen wend=`t'-1
377     reg dlnWeeklyEarnings l1dlnWeekHours l1dlnHourlyEarnings ///
378     if Date>=wstart & Date<=wend
379     replace nob=e(N) if Date==`t'
380     predict ptemp
381     replace pred=ptemp if Date==`t'
382     drop ptemp wstart wend
383   }
384   gen errsq=(pred-d.lnWeeklyEarnings)^2
385   summ errsq
386   scalar RWrmse`w'=r(mean)^.5
387   summ nob
388   scalar RWminobs`w'=r(min)
389   scalar RWmaxobs`w'=r(max)
390   drop errsq pred nob
391 }
392 scalar list
393 /*
394 RWmaxobs60 =          60
395 RWminobs60 =           2
396 RWrmse60 =   .06145693
397 */
398
399 quietly gsreg dlnWeeklyEarnings l1dlnWeeklyEarnings l2dlnWeeklyEarnings
l3dlnWeeklyEarnings ///
400   l4dlnWeeklyEarnings l5dlnWeeklyEarnings l6dlnWeeklyEarnings ///
401   l7dlnWeeklyEarnings l8dlnWeeklyEarnings l9dlnWeeklyEarnings l10dlnWeeklyEarnings
///
402   l11dlnWeeklyEarnings l12dlnWeeklyEarnings ///
403   l24dlnWeeklyEarnings l36dlnWeeklyEarnings ///
404   if tin(2011m1,2021m1), ///
405   ncomb(1,12) aic outsample(24) ///
406   samesample nindex( -1 aic -1 bic -1 rmse_out) results(gsreg_dlnWeeklyEarnings)
replace
407
408 reg d.lnWeeklyEarnings l3d.lnWeeklyEarnings l5d.lnWeeklyEarnings m1 m2 m3 m4 m5 m6
m7 m8 m9 m10 m11
409 scalar drop _all
410 quietly forval w=12(12)84 {

```

```

411 gen pred=.
412 gen nobs=.
413     forval t=620/733 {
414         gen wstart=`t'-'w'
415         gen wend=`t'-1
416         reg dlnWeeklyEarnings l3dlnWeeklyEarnings l5dlnWeeklyEarnings m1 m2 m3 m4 m5 m6 m7
417         if Date>=wstart & Date<=wend
418             replace nobs=e(N) if Date==`t'
419             predict ptemp
420             replace pred=ptemp if Date==`t'
421             drop ptemp wstart wend
422     }
423 gen errsq=(pred-d.lnWeeklyEarnings)^2
424 summ errsq
425 scalar RWrmse`w'=r(mean)^.5
426 summ nobs
427 scalar RWminobs`w'=r(min)
428 scalar RWmaxobs`w'=r(max)
429 drop errsq pred nobs
430 }
431 scalar list
432 /*
433 RWmaxobs84 =          84
434 RWminobs84 =           2
435 RWrmse84 =  .06004448
436 */
437
438 reg d.lnWeeklyEarnings l3d.lnWeeklyEarnings l5d.lnWeeklyEarnings
439 scalar drop _all
440 quietly forval w=12(12)84 {
441     gen pred=.
442     gen nobs=.
443         forval t=622/733 {
444             gen wstart=`t'-'w'
445             gen wend=`t'-1
446             reg dlnWeeklyEarnings l3dlnWeeklyEarnings l5dlnWeeklyEarnings l7dlnWeeklyEarnings
447             if Date>=wstart & Date<=wend
448                 replace nobs=e(N) if Date==`t'
449                 predict ptemp
450                 replace pred=ptemp if Date==`t'
451                 drop ptemp wstart wend
452             }
453 gen errsq=(pred-d.lnWeeklyEarnings)^2
454 summ errsq
455 scalar RWrmse`w'=r(mean)^.5
456 summ nobs
457 scalar RWminobs`w'=r(min)
458 scalar RWmaxobs`w'=r(max)

```

```

459 drop errsq pred nobis
460 }
461 scalar list
462 /*
463 RWmaxobs84 =      84
464 RWminobs84 =      2
465 RWrmse84 = .05250414
466 */
467
468 drop pflcount ub1 ub2 ub3 lb1 lb2 lb3
469
470 scalar rwormse = .05250414
471 reg d.lnWeeklyEarnings l(3,5,7)d.lnWeeklyEarnings if tin(,2021m2)
472 predict pd
473 gen pflcount=exp((rwormse^2)/2)*exp(1.lnWeeklyEarnings+pd) if Date==tm(2021m3)
474 gen ub1=exp((rwormse^2)/2)*exp(1.lnWeeklyEarnings+pd+1*rwormse) if Date==tm(2021m3)
475 gen lb1=exp((rwormse^2)/2)*exp(1.lnWeeklyEarnings+pd-1*rwormse) if Date==tm(2021m3)
476 gen ub2=exp((rwormse^2)/2)*exp(1.lnWeeklyEarnings+pd+2*rwormse) if Date==tm(2021m3)
477 gen lb2=exp((rwormse^2)/2)*exp(1.lnWeeklyEarnings+pd-2*rwormse) if Date==tm(2021m3)
478 gen ub3=exp((rwormse^2)/2)*exp(1.lnWeeklyEarnings+pd+3*rwormse) if Date==tm(2021m3)
479 gen lb3=exp((rwormse^2)/2)*exp(1.lnWeeklyEarnings+pd-3*rwormse) if Date==tm(2021m3)
480 drop pd
481
482 replace pflcount=WeeklyEarnings if Date==tm(2021m2)
483 replace ub1=WeeklyEarnings if Date==tm(2021m2)
484 replace ub2=WeeklyEarnings if Date==tm(2021m2)
485 replace ub3=WeeklyEarnings if Date==tm(2021m2)
486 replace lb1=WeeklyEarnings if Date==tm(2021m2)
487 replace lb2=WeeklyEarnings if Date==tm(2021m2)
488 replace lb3=WeeklyEarnings if Date==tm(2021m2)
489
490 twoway (tsrline ub3 ub2 if tin(2020m3,2021m3), ///
491   recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
492   (tsrline ub2 ub1 if tin(2020m3,2021m3), ///
493   recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
494   (tsrline ub1 pflcount if tin(2020m3,2021m3), ///
495   recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///
496   (tsrline pflcount lb1 if tin(2020m3,2021m3), ///
497   recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///
498   (tsrline lb1 lb2 if tin(2020m3,2021m3), ///
499   recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
500   (tsrline lb2 lb3 if tin(2020m3,2021m3), ///
501   recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
502   (tsline WeeklyEarnings pflcount if tin(2020m3,2021m3) , ///
503   lcolor(gs12 teal) lwidth(medthick medthick) ///
504   lpattern(solid longdash)) ///
505   (scatter withMarchEarnings Date if tin(2021m3,)), scheme(s1mono) legend(off)
506 graph export "WeeklyFan.png", replace
507
508 log close
509 translate "Final Project.smcl" "Final Project.txt", replace

```

## Appendix B

```

1      ____ (R)      ____  ____  ____  ____
2      /_  /  ____/  /  ____/
3      __/  /  /__/  /  /__/
4
5      Statistics/Data analysis
6
7      -----
8      --
9      name:  <unnamed>
10     log:   /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time
11     Series/Probl
12
13     > em Sets/Final Project/Final Project.smcl
14     log type:  smcl
15     opened on:  19 Apr 2021, 20:43:14
16
17     1 . import delimited "TS2020_Final_Project_Monthly.txt"
18       (6 vars, 375 obs)
19
20     2 . rename smul2455400500000001 Count
21
22     3 . rename smul2455400500000002 WeekHours
23
24     4 . rename smul2455400500000003 HourlyEarnings
25
26     5 . rename smul2455400500000011 WeeklyEarnings
27
28     6 . rename smul2455400800000001 ServiceCount
29
30     7 .
31     8 .
32     9 . label variable Count "Count"
33
34    10 . label variable WeekHours "WeekHours"
35
36    11 . label variable HourlyEarnings "HourlyEarnings"
37
38    12 . label variable WeeklyEarnings "WeeklyEarnings"
39
40    13 . label variable ServiceCount "ServiceCount"
41
42    14 .
43    15 .

```

```

40 16 . gen datec=date(date, "YMD")
41
42 17 . gen Date=mofd(datec)
43
44 18 . gen month=month(datec)
45
46 19 . format Date %tm
47
48 20 . tsset Date
49         time variable:  Date, 1990m1 to 2021m3
50         delta: 1 month
51
52 21 .
53 22 . gen withMarchCount = Count
54
55 23 . gen withMarchEarnings = WeeklyEarnings
56     (252 missing values generated)
57
58 24 . replace Count=. if tin(2021m3,)
59     (1 real change made, 1 to missing)
60
61 25 . replace WeeklyEarnings=. if tin(2021m3,)
62     (1 real change made, 1 to missing)
63
64 26 .
65 27 . gen lnCount = ln(Count)
66     (1 missing value generated)
67
68 28 . gen lnWeekHours = ln(WeekHours)
69     (252 missing values generated)
70
71 29 . gen lnHourlyEarnings = ln(HourlyEarnings)
72     (252 missing values generated)
73
74 30 . gen lnWeeklyEarnings = ln(WeeklyEarnings)
75     (253 missing values generated)
76
77 31 . gen lnServiceCount = ln(ServiceCount)
78
79 32 .
80 33 . gen m1=0
81
82 34 . replace m1=1 if month==1
83     (32 real changes made)
84
85 35 . gen m2=0
86
87 36 . replace m2=1 if month==2
88     (32 real changes made)
89
90 37 . gen m3=0

```

```
91
92     38 . replace m3=1 if month==3
93         (32 real changes made)
94
95     39 . gen m4=0
96
97     40 . replace m4=1 if month==4
98         (31 real changes made)
99
100    41 . gen m5=0
101
102    42 . replace m5=1 if month==5
103        (31 real changes made)
104
105    43 . gen m6=0
106
107    44 . replace m6=1 if month==6
108        (31 real changes made)
109
110    45 . gen m7=0
111
112    46 . replace m7=1 if month==7
113        (31 real changes made)
114
115    47 . gen m8=0
116
117    48 . replace m8=1 if month==8
118        (31 real changes made)
119
120    49 . gen m9=0
121
122    50 . replace m9=1 if month==9
123        (31 real changes made)
124
125    51 . gen m10=0
126
127    52 . replace m10=1 if month==10
128        (31 real changes made)
129
130    53 . gen m11=0
131
132    54 . replace m11=1 if month==11
133        (31 real changes made)
134
135    55 . gen m12=0
136
137    56 . replace m12=1 if month==12
138        (31 real changes made)
139
140    57 .
141    58 . gen dlnCount=d.lnCount
```

```
142         (2 missing values generated)
143
144 59 . gen l1dlnCount=l1d.lnCount
145     (2 missing values generated)
146
147 60 . gen l2dlnCount=l2d.lnCount
148     (3 missing values generated)
149
150 61 . gen l3dlnCount=l3d.lnCount
151     (4 missing values generated)
152
153 62 . gen l4dlnCount=l4d.lnCount
154     (5 missing values generated)
155
156 63 . gen l5dlnCount=l5d.lnCount
157     (6 missing values generated)
158
159 64 . gen l6dlnCount=l6d.lnCount
160     (7 missing values generated)
161
162 65 . gen l7dlnCount=l7d.lnCount
163     (8 missing values generated)
164
165 66 . gen l8dlnCount=l8d.lnCount
166     (9 missing values generated)
167
168 67 . gen l9dlnCount=l9d.lnCount
169     (10 missing values generated)
170
171 68 . gen l10dlnCount=l10d.lnCount
172     (11 missing values generated)
173
174 69 . gen l11dlnCount=l11d.lnCount
175     (12 missing values generated)
176
177 70 . gen l12dlnCount=l12d.lnCount
178     (13 missing values generated)
179
180 71 . gen l24dlnCount=l24d.lnCount
181     (25 missing values generated)
182
183 72 . gen l36dlnCount=l36d.lnCount
184     (37 missing values generated)
185
186 73 . gen l48dlnCount=l48d.lnCount
187     (49 missing values generated)
188
189 74 .
190 75 . gen dlnWeekHours=d.lnWeekHours
191     (253 missing values generated)
192
```

```

193 76 . gen l1dlnWeekHours=l1d.lnWeekHours
194     (254 missing values generated)
195
196 77 . gen l2dlnWeekHours=l2d.lnWeekHours
197     (255 missing values generated)
198
199 78 . gen l3dlnWeekHours=l3d.lnWeekHours
200     (256 missing values generated)
201
202 79 . gen l4dlnWeekHours=l4d.lnWeekHours
203     (257 missing values generated)
204
205 80 . gen l5dlnWeekHours=l5d.lnWeekHours
206     (258 missing values generated)
207
208 81 . gen l6dlnWeekHours=l6d.lnWeekHours
209     (259 missing values generated)
210
211 82 . gen l7dlnWeekHours=l7d.lnWeekHours
212     (260 missing values generated)
213
214 83 . gen l8dlnWeekHours=l8d.lnWeekHours
215     (261 missing values generated)
216
217 84 . gen l9dlnWeekHours=l9d.lnWeekHours
218     (262 missing values generated)
219
220 85 . gen l10dlnWeekHours=l10d.lnWeekHours
221     (263 missing values generated)
222
223 86 . gen l11dlnWeekHours=l11d.lnWeekHours
224     (264 missing values generated)
225
226 87 . gen l12dlnWeekHours=l12d.lnWeekHours
227     (265 missing values generated)
228
229 88 . gen l24dlnWeekHours=l24d.lnWeekHours
230     (277 missing values generated)
231
232 89 . gen l36dlnWeekHours=l36d.lnWeekHours
233     (289 missing values generated)
234
235 90 . gen l48dlnWeekHours=l48d.lnWeekHours
236     (301 missing values generated)
237
238 91 .
239 92 . gen dlnHourlyEarnings=d.lnHourlyEarnings
240     (253 missing values generated)
241
242 93 . gen l1dlnHourlyEarnings=l1d.lnHourlyEarnings
243     (254 missing values generated)

```



```
244
245     94 . gen l2dlnHourlyEarnings=l2d.lnHourlyEarnings
246         (255 missing values generated)
247
248     95 . gen l3dlnHourlyEarnings=l3d.lnHourlyEarnings
249         (256 missing values generated)
250
251     96 . gen l4dlnHourlyEarnings=l4d.lnHourlyEarnings
252         (257 missing values generated)
253
254     97 . gen l5dlnHourlyEarnings=l5d.lnHourlyEarnings
255         (258 missing values generated)
256
257     98 . gen l6dlnHourlyEarnings=l6d.lnHourlyEarnings
258         (259 missing values generated)
259
260     99 . gen l7dlnHourlyEarnings=l7d.lnHourlyEarnings
261         (260 missing values generated)
262
263    100 . gen l8dlnHourlyEarnings=l8d.lnHourlyEarnings
264         (261 missing values generated)
265
266    101 . gen l9dlnHourlyEarnings=l9d.lnHourlyEarnings
267         (262 missing values generated)
268
269    102 . gen l10dlnHourlyEarnings=l10d.lnHourlyEarnings
270         (263 missing values generated)
271
272    103 . gen l11dlnHourlyEarnings=l11d.lnHourlyEarnings
273         (264 missing values generated)
274
275    104 . gen l12dlnHourlyEarnings=l12d.lnHourlyEarnings
276         (265 missing values generated)
277
278    105 . gen l24dlnHourlyEarnings=l24d.lnHourlyEarnings
279         (277 missing values generated)
280
281    106 . gen l36dlnHourlyEarnings=l36d.lnHourlyEarnings
282         (289 missing values generated)
283
284    107 . gen l48dlnHourlyEarnings=l48d.lnHourlyEarnings
285         (301 missing values generated)
286
287    108 .
288    109 . gen dlnWeeklyEarnings=d.lnWeeklyEarnings
289         (254 missing values generated)
290
291    110 . gen l1dlnWeeklyEarnings=l1d.lnWeeklyEarnings
292         (254 missing values generated)
293
294    111 . gen l2dlnWeeklyEarnings=l2d.lnWeeklyEarnings
```

```

295         (255 missing values generated)
296
297     112 . gen l3dlnWeeklyEarnings=l3d.lnWeeklyEarnings
298         (256 missing values generated)
299
300     113 . gen l4dlnWeeklyEarnings=l4d.lnWeeklyEarnings
301         (257 missing values generated)
302
303     114 . gen l5dlnWeeklyEarnings=l5d.lnWeeklyEarnings
304         (258 missing values generated)
305
306     115 . gen l6dlnWeeklyEarnings=l6d.lnWeeklyEarnings
307         (259 missing values generated)
308
309     116 . gen l7dlnWeeklyEarnings=l7d.lnWeeklyEarnings
310         (260 missing values generated)
311
312     117 . gen l8dlnWeeklyEarnings=l8d.lnWeeklyEarnings
313         (261 missing values generated)
314
315     118 . gen l9dlnWeeklyEarnings=l9d.lnWeeklyEarnings
316         (262 missing values generated)
317
318     119 . gen l10dlnWeeklyEarnings=l10d.lnWeeklyEarnings
319         (263 missing values generated)
320
321     120 . gen l11dlnWeeklyEarnings=l11d.lnWeeklyEarnings
322         (264 missing values generated)
323
324     121 . gen l12dlnWeeklyEarnings=l12d.lnWeeklyEarnings
325         (265 missing values generated)
326
327     122 . gen l24dlnWeeklyEarnings=l24d.lnWeeklyEarnings
328         (277 missing values generated)
329
330     123 . gen l36dlnWeeklyEarnings=l36d.lnWeeklyEarnings
331         (289 missing values generated)
332
333     124 . gen l48dlnWeeklyEarnings=l48d.lnWeeklyEarnings
334         (301 missing values generated)
335
336     125 .
337     126 . gen dlnServiceCount=d.lnServiceCount
338         (1 missing value generated)
339
340     127 . gen l1dlnServiceCount=l1d.lnServiceCount
341         (2 missing values generated)
342
343     128 . gen l2dlnServiceCount=l2d.lnServiceCount
344         (3 missing values generated)
345

```

```

346 129 . gen l3dlnServiceCount=l3d.lnServiceCount
347     (4 missing values generated)
348
349 130 . gen l4dlnServiceCount=l4d.lnServiceCount
350     (5 missing values generated)
351
352 131 . gen l5dlnServiceCount=l5d.lnServiceCount
353     (6 missing values generated)
354
355 132 . gen l6dlnServiceCount=l6d.lnServiceCount
356     (7 missing values generated)
357
358 133 . gen l7dlnServiceCount=l7d.lnServiceCount
359     (8 missing values generated)
360
361 134 . gen l8dlnServiceCount=l8d.lnServiceCount
362     (9 missing values generated)
363
364 135 . gen l9dlnServiceCount=l9d.lnServiceCount
365     (10 missing values generated)
366
367 136 . gen l10dlnServiceCount=l10d.lnServiceCount
368     (11 missing values generated)
369
370 137 . gen l11dlnServiceCount=l11d.lnServiceCount
371     (12 missing values generated)
372
373 138 . gen l12dlnServiceCount=l12d.lnServiceCount
374     (13 missing values generated)
375
376 139 . gen l24dlnServiceCount=l24d.lnServiceCount
377     (25 missing values generated)
378
379 140 . gen l36dlnServiceCount=l36d.lnServiceCount
380     (37 missing values generated)
381
382 141 . gen l48dlnServiceCount=l48d.lnServiceCount
383     (49 missing values generated)
384
385 142 .
386 143 . /*
387     > The project is to forecast the March non-seasonally adjusted estimates of
ave
388     > rage weekly earnings and total employment for private employers (total
privat
389     > e) for a Florida MSA of your choice and write up a professional report on
you
390     > r forecast.
391     > */
392 144 . /* Count and WeeklyEarnings */
393 145 .

```

```

394 146 . summ Count WeekHours HourlyEarnings WeeklyEarnings ServiceCount
395
396      Variable |           Obs      Mean   Std. Dev.      Min      Max
397  -----+-----
398      Count |           374    14.18556    6.880684      5.3      28
399      WeekHours |           123    36.88455    3.791817     28.3     45.8
400      HourlyEarn~s |           123      19.72    2.903968     15.01     24.6
401      WeeklyEarn~s |           122   719.6542    84.57241    503.79    916.1
402      ServiceCount |           375    10.43387    5.959179      3.9     22.8
403
404 147 . summ lnCount lnWeekHours lnHourlyEarnings lnWeeklyEarnings lnServiceCount
405
406      Variable |           Obs      Mean   Std. Dev.      Min      Max
407  -----+-----
408      lnCount |           374      2.5174    .5398403    1.667707    3.332205
409      lnWeekHours |           123      3.602488    .10385    3.342862    3.824284
410      lnHourlyEa~s |           123      2.970779    .1482819    2.708717    3.202746
411      lnWeeklyEa~s |           122      6.571775    .1195694    6.222159    6.820126
412      lnServiceC~t |           375      2.172053    .5985689    1.360977    3.12676
413
414 148 .
415 149 . ac lnCount, saving(lnCount_ac, replace)
416      (file lnCount_ac.gph saved)
417
418 150 . pac lnCount, saving(lnCount_pac, replace)
419      (file lnCount_pac.gph saved)
420
421 151 . graph combine lnCount_ac.gph lnCount_pac.gph, saving(lnCount_ac_pac,
replace)
422      (file lnCount_ac_pac.gph saved)
423
424 152 . graph export "lnCount_ac_pac.png", replace
425      (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
426      > /Final Project/lnCount_ac_pac.png written in PNG format)
427
428 153 . ** Probably need to difference
429 154 .
430 155 . ac lnWeeklyEarnings, saving(lnWeeklyEarnings_ac, replace)
431      (file lnWeeklyEarnings_ac.gph saved)
432
433 156 . pac lnWeeklyEarnings, saving(lnWeeklyEarnings_pac, replace)
434      (file lnWeeklyEarnings_pac.gph saved)
435
436 157 . graph combine lnWeeklyEarnings_ac.gph lnWeeklyEarnings_pac.gph,
saving(lnWeek
437      > lyEarnings_ac_pac, replace)
438      (file lnWeeklyEarnings_ac_pac.gph saved)
439
440 158 . graph export "lnWeeklyEarnings_ac_pac.png", replace

```

```

441 (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
442 > /Final Project/lnWeeklyEarnings_ac_pac.png written in PNG format)
443
444 159 . ** Probably need to difference
445 160 .
446 161 . *starter models for count
447 162 . *I used a pair plot to examine the rise and fall of variables with respect
to
448 > each other
449 163 . reg d.lnCount l(12,24,36,48)d.lnCount // .01637
450
451 Source | SS df MS Number of obs =
325
452 -----+----- F(4, 320) =
16.54
453 Model | .017539188 4 .004384797 Prob > F =
0.0000
454 Residual | .084856979 320 .000265178 R-squared =
0.1713
455 -----+----- Adj R-squared =
0.1609
456 Total | .102396167 324 .000316038 Root MSE =
.01628
457
458 -----
-
459 D.lnCount | Coef. Std. Err. t P>|t| [95% Conf.
Interval]
460 -----+-----
-
461 lnCount |
462 L12D. | .3609966 .0621085 5.81 0.000 .238804
.4831893
463 L24D. | .137848 .0617615 2.23 0.026 .016338
.259358
464 L36D. | -.0160136 .0614584 -0.26 0.795 -.1369272
.1049
465 L48D. | .1265117 .0585322 2.16 0.031 .0113551
.2416683
466 |
467 _cons | .0017116 .0009853 1.74 0.083 -.0002269
.0036502
468 -----
-
469
470 164 . scalar drop _all
471
472 165 . quietly forval w=12(12)144 {
473
474 166 . scalar list

```

```

475     RWmaxobs144 =      144
476     RWminobs144 =      12
477     RWrmse144 =    .0172276
478     RWmaxobs132 =      132
479     RWminobs132 =      12
480     RWrmse132 =    .0172128
481     RWmaxobs120 =      120
482     RWminobs120 =      12
483     RWrmse120 =    .01721825
484     RWmaxobs108 =      108
485     RWminobs108 =      12
486     RWrmse108 =    .01723674
487     RWmaxobs96 =       96
488     RWminobs96 =      12
489     RWrmse96 =    .01722006
490     RWmaxobs84 =       84
491     RWminobs84 =      12
492     RWrmse84 =    .01726063
493     RWmaxobs72 =       72
494     RWminobs72 =      12
495     RWrmse72 =    .01722377
496     RWmaxobs60 =       60
497     RWminobs60 =      12
498     RWrmse60 =    .0173443
499     RWmaxobs48 =       48
500     RWminobs48 =      12
501     RWrmse48 =    .01755803
502     RWmaxobs36 =       36
503     RWminobs36 =      12
504     RWrmse36 =    .01805924
505     RWmaxobs24 =       24
506     RWminobs24 =      12
507     RWrmse24 =    .0185871
508     RWmaxobs12 =       12
509     RWminobs12 =      12
510     RWrmse12 =    .02320505
511
512     167 . /*
513         > RWmaxobs132 =      132
514         > RWminobs132 =      12
515         > RWrmse132 =    .0172128
516         > */
517     168 .
518     169 . reg d.lnCount l(5,12,24,36,48)d.lnCount l(5)d.lnWeekHours m5 // .01711
519
520           Source |           SS           df           MS       Number of obs   =
116
521     -----+-----
5.94
522           Model |    .012171566           7    .001738795       Prob > F           =
0.0000

```

```

523      Residual |      .03162877      108      .000292859      R-squared      =
0.2779
524      -----+-----
0.2311
525      Total |      .043800336      115      .000380872      Root MSE      =
.01711
526
527      -----
-
528      D.lnCount |      Coef.      Std. Err.      t      P>|t|      [95% Conf.
Interval]
529      -----+-----
-
530      lnCount |
531      L5D. |      -.1231921      .0845717      -1.46      0.148      -.290828
.0444438
532      L12D. |      .5811114      .1685831      3.45      0.001      .2469504
.9152724
533      L24D. |      -.1196017      .1627467      -0.73      0.464      -.4421938
.2029904
534      L36D. |      .2532303      .1742525      1.45      0.149      -.0921684
.5986291
535      L48D. |      .1341638      .1858633      0.72      0.472      -.2342495
.5025771
536
537      lnWeekHours |
538      L5D. |      .0170123      .0364906      0.47      0.642      -.0553184
.089343
539
540      m5 |      .0067588      .0061605      1.10      0.275      -.0054524
.0189699
541      _cons |      .0004279      .0018229      0.23      0.815      -.0031854
.0040412
542      -----
-
543
544      170 . scalar drop _all
545
546      171 . quietly forval w=12(12)84 {
547
548      172 . scalar list
549      RWmaxobs84 =      84
550      RWminobs84 =      23
551      RWrmse84 =      .01950911
552      RWmaxobs72 =      72
553      RWminobs72 =      23
554      RWrmse72 =      .01949719
555      RWmaxobs60 =      60
556      RWminobs60 =      23
557      RWrmse60 =      .0199438
558      RWmaxobs48 =      48

```

```

559     RWminobs48 =      23
560     RWrmse48 = .02035982
561     RWmaxobs36 =      36
562     RWminobs36 =      23
563     RWrmse36 = .02138785
564     RWmaxobs24 =      24
565     RWminobs24 =      23
566     RWrmse24 = .02268585
567     RWmaxobs12 =      12
568     RWminobs12 =      12
569     RWrmse12 = .05004898
570
571     173 . /*
572         > RWmaxobs84 =      84
573         > RWminobs84 =      23
574         > RWrmse84 = .01950911
575         > */
576     174 .
577     175 . quietly gsreg dlnCount l1dlnCount l2dlnCount l3dlnCount l4dlnCount
15dlnCount
578         > l6dlnCount ///
579         >         l7dlnCount l8dlnCount l9dlnCount l10dlnCount l11dlnCount
l12dlnCount
580         > ///
581         >         l24dlnCount l36dlnCount l48dlnCount ///
582         >         if tin(1990m1,2021m1), ///
583         >         ncomb(1,12) aic outsample(24) fix(m1 m2 m3 m4 m5 m6 m7 m8 m9 m10
m11
584         > m12) ///
585         >         samesample nindex( -1 aic -1 bic -1 rmse_out)
results(gsreg_dlnCount)
586         > replace
587
588     176 .
589     177 . *gsreg suggestions
590     178 . reg d.lnCount l12d.lnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11
591
592             Source |          SS          df          MS      Number of obs   =
361
593     -----+-----
6.91
594             Model |    .022616974          12    .001884748      Prob > F           =
0.0000
595             Residual |    .094871179         348    .000272618      R-squared          =
0.1925
596     -----+-----
0.1647
597             Total |    .117488153         360    .000326356      Root MSE          =
.01651
598

```



```

599 -----
600 -
601 D.lnCount | Coef. Std. Err. t P>|t| [95% Conf.
602 Interval]
603 -----+-----
604 -
605 lnCount |
606 L12D. | .1748571 .0594184 2.94 0.003 .0579928
607 .2917215
608 |
609 m1 | -.0099477 .0043004 -2.31 0.021 -.0184056
610 -.0014897
611 m2 | .0009939 .0042297 0.23 0.814 -.0073251
612 .0093129
613 m3 | .0030247 .0042759 0.71 0.480 -.0053851
614 .0114345
615 m4 | -.0071933 .0042648 -1.69 0.093 -.0155814
616 .0011948
617 m5 | -.0098194 .0043178 -2.27 0.024 -.0183118
618 -.0013271
619 m6 | -.0133285 .0043874 -3.04 0.003 -.0219576
620 -.0046994
621 m7 | -.0091828 .0042967 -2.14 0.033 -.0176336
622 -.000732
623 m8 | -.0017998 .0042632 -0.42 0.673 -.0101846
624 .006585
625 m9 | -.006737 .0042824 -1.57 0.117 -.0151597
626 .0016858
627 m10 | .0062149 .0042795 1.45 0.147 -.0022021
628 .0146319
629 m11 | .0042124 .0042811 0.98 0.326 -.0042078
630 .0126325
631 _cons | .0072199 .0030452 2.37 0.018 .0012306
632 .0132093
633 -----
634 -
635
636 179 . scalar drop _all
637
638 180 . quietly forval w=12(12)144 {
639
640 181 . scalar list
641 RWmaxobs144 = 144
642 RWminobs144 = 12
643 RWrmse144 = .01824906
644 RWmaxobs132 = 132
645 RWminobs132 = 12
646 RWrmse132 = .01832173
647 RWmaxobs120 = 120
648 RWminobs120 = 12
649 RWrmse120 = .01833557

```

```

633      RWmaxobs108 =      108
634      RWminobs108 =      12
635      RWrmse108 = .01841089
636      RWmaxobs96 =      96
637      RWminobs96 =      12
638      RWrmse96 = .01836974
639      RWmaxobs84 =      84
640      RWminobs84 =      12
641      RWrmse84 = .01849267
642      RWmaxobs72 =      72
643      RWminobs72 =      12
644      RWrmse72 = .01861349
645      RWmaxobs60 =      60
646      RWminobs60 =      12
647      RWrmse60 = .01911515
648      RWmaxobs48 =      48
649      RWminobs48 =      12
650      RWrmse48 = .01922268
651      RWmaxobs36 =      36
652      RWminobs36 =      12
653      RWrmse36 = .01991683
654      RWmaxobs24 =      24
655      RWminobs24 =      12
656      RWrmse24 = .02022186
657      RWmaxobs12 =      12
658      RWminobs12 =      12
659      RWrmse12 = .02009249
660
661      182 . /*
662          > RWmaxobs144 =      144
663          > RWminobs144 =      12
664          > RWrmse144 = .01824906
665          > */
666      183 .
667      184 . reg d.lnCount l(12,36)d.lnCount m1 m2 m3 m4 m5 m6 m7 m8 m9 m10 m11
668
669          Source |           SS           df           MS      Number of obs      =
337
670      -----+-----
5.68
671          Model |   .019946057           13   .001534312      Prob > F           =
0.0000
672          Residual |   .087185203          323   .000269923      R-squared           =
0.1862
673      -----+-----
0.1534
674          Total |   .107131259          336   .000318843      Root MSE           =
.01643
675
676      -----
-
```

```

677      D.lnCount |      Coef.   Std. Err.      t    P>|t|      [95% Conf.
678      Interval]
679      -----+-----
680      lnCount |
681      L12D. |   .1849403   .0636401    2.91   0.004    .059739
682      .3101417
683      L36D. |  -.049332   .0606582   -0.81   0.417   -.1686671
684      .0700031
685      |
686      m1 |  -.0073418   .0044769   -1.64   0.102   -.0161493
687      .0014658
688      m2 |   .0022711   .0043559    0.52   0.602   -.0062984
689      .0108407
690      m3 |   .0043593   .004416    0.99   0.324   -.0043285
691      .0130471
692      m4 |  -.0065438   .0043922   -1.49   0.137   -.0151847
693      .002097
694      m5 |  -.0089215   .0045194   -1.97   0.049   -.0178126
695      -.0000304
696      m6 |  -.0133453   .0046241   -2.89   0.004   -.0224425
697      -.004248
698      m7 |  -.0085154   .004457   -1.91   0.057   -.0172839
699      .0002531
700      m8 |  -.0004554   .004392   -0.10   0.917   -.0090959
701      .0081852
702      m9 |  -.0056625   .0044299   -1.28   0.202   -.0143775
703      .0030526
704      m10 |   .0071688   .0044386    1.62   0.107   -.0015635
705      .0159011
706      m11 |   .0042074   .0044259    0.95   0.343   -.0044998
707      .0129146
708      _cons |   .0067355   .0031722    2.12   0.034    .0004948
709      .0129762
710      -----+-----
711      -
712
713      185 . scalar drop _all
714
715      186 . quietly forval w=12(12)144 {
716
717      187 . scalar list
718      RWmaxobs144 =      144
719      RWminobs144 =      12
720      RWrmse144 =   .01777071
721      RWmaxobs132 =      132
722      RWminobs132 =      12
723      RWrmse132 =   .01782557
724      RWmaxobs120 =      120
725      RWminobs120 =      12
726      RWrmse120 =   .01785253

```

```

711      RWmaxobs108 =      108
712      RWminobs108 =      12
713      RWrmse108 = .01794692
714      RWmaxobs96 =      96
715      RWminobs96 =      12
716      RWrmse96 = .01793358
717      RWmaxobs84 =      84
718      RWminobs84 =      12
719      RWrmse84 = .01803355
720      RWmaxobs72 =      72
721      RWminobs72 =      12
722      RWrmse72 = .01807408
723      RWmaxobs60 =      60
724      RWminobs60 =      12
725      RWrmse60 = .01843535
726      RWmaxobs48 =      48
727      RWminobs48 =      12
728      RWrmse48 = .01835092
729      RWmaxobs36 =      36
730      RWminobs36 =      12
731      RWrmse36 = .01863303
732      RWmaxobs24 =      24
733      RWminobs24 =      12
734      RWrmse24 = .0196745
735      RWmaxobs12 =      12
736      RWminobs12 =      12
737      RWrmse12 = .01880291
738
739      188 . /*
740          > RWmaxobs144 =      144
741          > RWminobs144 =      12
742          > RWrmse144 = .01777071
743          > */
744      189 .
745      190 . reg d.lnCount l4d.lnWeekHours l9d.lnWeekHours l8d.lnHourlyEarnings m1 m2
m3 m
746          > 4 m5 m6 m7 m8 m9 m10 m11
747
748          Source |      SS      df      MS      Number of obs      =
112
749      -----+-----
3.24
750          Model | .013917393      14      .0009941      Prob > F      =
0.0003
751          Residual | .029798432      97      .0003072      R-squared      =
0.3184
752      -----+-----
0.2200
753          Total | .043715825     111      .000393836      Root MSE      =
.01753
754

```

755	-----						
756	--						
757	---	D.lnCount		Coef.	Std. Err.	t	P> t  [95% Conf.
758	Interv						
759	> al]						
760	-----+-----						
761	--						
762	---	lnWeekHours					
763		L4D.		-.0013847	.0384012	-0.04	0.971 -.0776005
764	.074						
765	> 831						
766		L9D.		.0397686	.0385964	1.03	0.305 -.0368346
767	.1163						
768	> 718						
769							
770		lnHourlyEarnings					
771		L8D.		-.039029	.0414024	-0.94	0.348 -.1212014
772	.0431						
773	> 433						
774							
775		m1		-.0097045	.0078517	-1.24	0.219 -.0252879
776	.0058						
777	> 789						
778		m2		.0000949	.0079445	0.01	0.990 -.0156727
779	.0158						
780	> 626						
781		m3		-.004712	.0083585	-0.56	0.574 -.0213013
782	.0118						
783	> 773						
784		m4		-.0273667	.0081729	-3.35	0.001 -.0435876
785	-.0111						
786	> 459						
787		m5		-.0076836	.0081259	-0.95	0.347 -.0238112
788	.008						
789	> 444						
790		m6		-.020254	.0081465	-2.49	0.015 -.0364227
791	-.0040						
792	> 854						
793		m7		-.0130812	.0081852	-1.60	0.113 -.0293265
794	.0031						
795	> 642						
796		m8		.0041701	.0081051	0.51	0.608 -.0119164
797	.0202						
798	> 565						
799		m9		-.0089171	.0082764	-1.08	0.284 -.0253435
800	.0075						
801	> 093						
802		m10		.0153608	.0081153	1.89	0.061 -.0007459
803	.0314						

```

790         > 674
791             m11 |      .0040463      .0079619      0.51      0.612      -.0117559
.0198
792         > 485
793             _cons |    .0094122    .0056462      1.67      0.099      -.0017939
.0206
794         > 183
795         -----
--
796         ---
797
798     191 . scalar drop _all
799
800     192 . quietly forval w=12(12)84 {
801
802     193 . scalar list
803         RWmaxobs84 =          84
804         RWminobs84 =           2
805         RWrmse84 =   .01847546
806         RWmaxobs72 =          72
807         RWminobs72 =           2
808         RWrmse72 =   .01855448
809         RWmaxobs60 =          60
810         RWminobs60 =           2
811         RWrmse60 =   .01850723
812         RWmaxobs48 =          48
813         RWminobs48 =           2
814         RWrmse48 =   .01850217
815         RWmaxobs36 =          36
816         RWminobs36 =           2
817         RWrmse36 =   .01942535
818         RWmaxobs24 =          24
819         RWminobs24 =           2
820         RWrmse24 =   .02208272
821         RWmaxobs12 =          12
822         RWminobs12 =           2
823         RWrmse12 =   .0176238
824
825     194 . /*
826         > RWmaxobs12 =          12
827         > RWminobs12 =           2
828         > RWrmse12 =   .0176238
829         > */
830     195 .
831     196 . scalar rwrms = .0172128
832
833     197 . reg d.lnCount l(12,24,36,48)d.lnCount if tin(,2021m2)
834
835             Source |           SS           df           MS      Number of obs      =

```

```

836 -----+----- F(4, 320) =
16.54
837 Model | .017539188 4 .004384797 Prob > F =
0.0000
838 Residual | .084856979 320 .000265178 R-squared =
0.1713
839 -----+----- Adj R-squared =
0.1609
840 Total | .102396167 324 .000316038 Root MSE =
.01628
841
842 -----
-
843 D.lnCount | Coef. Std. Err. t P>|t| [95% Conf.
Interval]
844 -----+-----
-
845 lnCount |
846 L12D. | .3609966 .0621085 5.81 0.000 .238804
.4831893
847 L24D. | .137848 .0617615 2.23 0.026 .016338
.259358
848 L36D. | -.0160136 .0614584 -0.26 0.795 -.1369272
.1049
849 L48D. | .1265117 .0585322 2.16 0.031 .0113551
.2416683
850 |
851 _cons | .0017116 .0009853 1.74 0.083 -.0002269
.0036502
852 -----
-
853
854 198 . predict pd
855 (option xb assumed; fitted values)
856 (49 missing values generated)
857
858 199 . gen pflcount=exp((rwrmsse^2)/2)*exp(1.lnCount+pd) if Date==tm(2021m3)
859 (374 missing values generated)
860
861 200 . gen ub1=exp((rwrmsse^2)/2)*exp(1.lnCount+pd+1*rwrmsse) if Date==tm(2021m3)
862 (374 missing values generated)
863
864 201 . gen lb1=exp((rwrmsse^2)/2)*exp(1.lnCount+pd-1*rwrmsse) if Date==tm(2021m3)
865 (374 missing values generated)
866
867 202 . gen ub2=exp((rwrmsse^2)/2)*exp(1.lnCount+pd+2*rwrmsse) if Date==tm(2021m3)
868 (374 missing values generated)
869
870 203 . gen lb2=exp((rwrmsse^2)/2)*exp(1.lnCount+pd-2*rwrmsse) if Date==tm(2021m3)
871 (374 missing values generated)
872

```

```

873 204 . gen ub3=exp((rwrmse^2)/2)*exp(1.lnCount+pd+3*rwrmse) if Date==tm(2021m3)
874      (374 missing values generated)
875
876 205 . gen lb3=exp((rwrmse^2)/2)*exp(1.lnCount+pd-3*rwrmse) if Date==tm(2021m3)
877      (374 missing values generated)
878
879 206 . drop pd
880
881 207 .
882 208 . replace pflcount=Count if Date==tm(2021m2)
883      (1 real change made)
884
885 209 . replace ub1=Count if Date==tm(2021m2)
886      (1 real change made)
887
888 210 . replace ub2=Count if Date==tm(2021m2)
889      (1 real change made)
890
891 211 . replace ub3=Count if Date==tm(2021m2)
892      (1 real change made)
893
894 212 . replace lb1=Count if Date==tm(2021m2)
895      (1 real change made)
896
897 213 . replace lb2=Count if Date==tm(2021m2)
898      (1 real change made)
899
900 214 . replace lb3=Count if Date==tm(2021m2)
901      (1 real change made)
902
903 215 .
904 216 . twoway (tsrline ub3 ub2 if tin(2020m3,2021m3), ///
905      > recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
906      > (tsrline ub2 ub1 if tin(2020m3,2021m3), ///
907      > recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
908      > (tsrline ub1 pflcount if tin(2020m3,2021m3), ///
909      > recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///
910      > (tsrline pflcount lb1 if tin(2020m3,2021m3), ///
911      > recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///
912      > (tsrline lb1 lb2 if tin(2020m3,2021m3), ///
913      > recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
914      > (tsrline lb2 lb3 if tin(2020m3,2021m3), ///
915      > recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
916      > (tsline Count pflcount if tin(2020m3,2021m3) , ///
917      > lcolor(gs12 teal) lwidth(medthick medthick) ///
918      > lpattern(solid longdash)) ///
919      > (scatter withMarchCount Date if tin(2021m3,)), scheme(slmono)
legend(
920      > off)
921
922 217 . graph export "CountFan.png", replace

```



```

923      (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
924      > /Final Project/CountFan.png written in PNG format)
925
926      218 .
927      219 . /*-----
---
928      > -*/
929      220 .
930      221 . *starter models for weekly earnings
931      222 . reg d.lnWeeklyEarnings l1d.lnWeekHours ld.lnHourlyEarnings
932
933      Source |      SS      df      MS      Number of obs      =
120
934      -----+----- F(2, 117)      =
1.48
935      Model |      .0071986      2      .0035993      Prob > F      =
0.2316
936      Residual |      .284290844      117      .002429836      R-squared      =
0.0247
937      -----+----- Adj R-squared      =
0.0080
938      Total |      .291489443      119      .002449491      Root MSE      =
.04929
939
940      -----
--
941      ---
942      D. |
943      lnWeeklyEarnings |      Coef.      Std. Err.      t      P>|t|      [95% Conf.
Interv
944      > al]
945      -----+-----
--
946      ---
947      lnWeekHours |
948      LD. |      -.1334776      .1058563      -1.26      0.210      -.3431204
.0761
949      > 651
950      |
951      lnHourlyEarnings |
952      LD. |      .0746807      .1166933      0.64      0.523      -.1564244
.3057
953      > 857
954      |
955      _cons |      .0015108      .0045071      0.34      0.738      -.0074152
.0104
956      > 368
957      -----
--
958      ---

```

```

959
960 223 . scalar drop _all
961
962 224 . quietly forval w=12(12)84 {
963
964 225 . scalar list
965     RWmaxobs84 =      84
966     RWminobs84 =       2
967     RWrmse84 = .06183191
968     RWmaxobs72 =      72
969     RWminobs72 =       2
970     RWrmse72 = .06162109
971     RWmaxobs60 =      60
972     RWminobs60 =       2
973     RWrmse60 = .06144232
974     RWmaxobs48 =      48
975     RWminobs48 =       2
976     RWrmse48 = .0618403
977     RWmaxobs36 =      36
978     RWminobs36 =       2
979     RWrmse36 = .06201409
980     RWmaxobs24 =      24
981     RWminobs24 =       2
982     RWrmse24 = .06224974
983     RWmaxobs12 =      12
984     RWminobs12 =       2
985     RWrmse12 = .06583082
986
987 226 . /*
988     > RWmaxobs60 =      60
989     > RWminobs60 =       2
990     > RWrmse60 = .06145693
991     > */
992 227 .
993 228 . quietly gsreg dlnWeeklyEarnings l1dlnWeeklyEarnings l2dlnWeeklyEarnings
l3dln
994     > WeeklyEarnings ///
995     >         l4dlnWeeklyEarnings l5dlnWeeklyEarnings l6dlnWeeklyEarnings ///
996     >         l7dlnWeeklyEarnings l8dlnWeeklyEarnings l9dlnWeeklyEarnings
l10dlnWee
997     > klyEarnings ///
998     >         l11dlnWeeklyEarnings l12dlnWeeklyEarnings ///
999     >         l24dlnWeeklyEarnings l36dlnWeeklyEarnings ///
1000     >         if tin(2011m1,2021m1), ///
1001     >         ncomb(1,12) aic outsample(24) ///
1002     >         samesample nindex( -1 aic -1 bic -1 rmse_out)
results(gsreg_dlnWeekly
1003     > Earnings) replace
1004
1005 229 .

```

```

1006      230 . reg d.lnWeeklyEarnings l3d.lnWeeklyEarnings l5d.lnWeeklyEarnings m1 m2 m3
1007      m4
1008      > m5 m6 m7 m8 m9 m10 m11
1009
1010      Source |          SS          df           MS      Number of obs   =
1011      116
1012      -----+-----
1013      2.16
1014      Model |    .061304493          13    .00471573   Prob > F           =
1015      0.0166
1016      Residual |    .222983103         102    .002186109   R-squared          =
1017      0.2156
1018      -----+-----
1019      0.1157
1020      Total |    .284287596         115    .002472066   Root MSE          =
1021      .04676
1022
1023      -----
1024      --
1025      ---
1026      D.          |
1027      lnWeeklyEarnings |      Coef.   Std. Err.      t    P>|t|     [95% Conf.
1028      Interv
1029      > al]
1030      -----+-----
1031      --
1032      ---
1033      lnWeeklyEarnings |
1034      L3D. |    -.2267216    .0950679    -2.38   0.019   -.4152883
1035      -.0381
1036      > 549
1037      L5D. |    -.1621197    .095104    -1.70   0.091   -.350758
1038      .0265
1039      > 186
1040      |
1041      m1 |    -.013308    .0213233    -0.62   0.534   -.0556027
1042      .0289
1043      > 866
1044      m2 |     .020775    .0212478     0.98   0.331   -.02137
1045      .06
1046      > 292
1047      m3 |    -.0123903    .0220875    -0.56   0.576   -.0562008
1048      .0314
1049      > 201
1050      m4 |     .0105198    .0219037     0.48   0.632   -.0329261
1051      .0539
1052      > 657
1053      m5 |     .0377285    .0216445     1.74   0.084   -.0052032
1054      .0806
1055      > 602

```

```

1039          m6 |      .0272631   .0216181   1.26   0.210   -.0156164
      .0701
1040      > 426
1041          m7 |     -.0220653   .0214504   -1.03   0.306   -.064612
      .0204
1042      > 813
1043          m8 |      .0152172   .0210597   0.72   0.472   -.0265547
      .0569
1044      > 891
1045          m9 |      .0201901   .0215988   0.93   0.352   -.0226509
      .0630
1046      > 312
1047          m10 |     .0207722   .021844   0.95   0.344   -.0225553
      .0640
1048      > 997
1049          m11 |     .0084712   .0217091   0.39   0.697   -.0345888
      .0515
1050      > 312
1051      _cons |     -.0073031   .0151547   -0.48   0.631   -.0373625
      .0227
1052      > 563
1053      -----
      --
1054      ---
1055
1056      231 . scalar drop _all
1057
1058      232 . quietly forval w=12(12)84 {
1059
1060      233 . scalar list
1061          RWmaxobs84 =      84
1062          RWminobs84 =       2
1063          RWrmse84 = .06011868
1064          RWmaxobs72 =      72
1065          RWminobs72 =       2
1066          RWrmse72 = .06057642
1067          RWmaxobs60 =      60
1068          RWminobs60 =       2
1069          RWrmse60 = .06071208
1070          RWmaxobs48 =      48
1071          RWminobs48 =       2
1072          RWrmse48 = .06042055
1073          RWmaxobs36 =      36
1074          RWminobs36 =       2
1075          RWrmse36 = .06125152
1076          RWmaxobs24 =      24
1077          RWminobs24 =       2
1078          RWrmse24 = .06537943
1079          RWmaxobs12 =      12
1080          RWminobs12 =       2
1081          RWrmse12 = .0702019

```

```

1082
1083     234 . /*
1084         > RWmaxobs84 =      84
1085         > RWminobs84 =      2
1086         > RWrmse84 = .06004448
1087         > */
1088     235 .
1089     236 . reg d.lnWeeklyEarnings l3d.lnWeeklyEarnings l5d.lnWeeklyEarnings
1090           17d.lnWeekly
1091           > Earnings
1092
1093           Source |      SS      df      MS      Number of obs      =
1094           114
1095           -----+-----
1096           3.85
1097           Model |   .026396014      3   .008798671   Prob > F      =
1098           0.0115
1099           Residual |   .251283445     110   .002284395   R-squared      =
1100           0.0951
1101           -----+-----
1102           0.0704
1103           Total |   .277679459     113   .00245734   Root MSE      =
1104           .0478
1105
1106           -----
1107           --
1108           ---
1109           D. |
1110           lnWeeklyEarnings |      Coef.   Std. Err.      t    P>|t|     [95% Conf.
1111           Interv
1112           > al]
1113           -----+-----
1114           --
1115           ---
1116           lnWeeklyEarnings |
1117           L3D. |   -.2408947   .0906673    -2.66   0.009   -.4205761
1118           -.0612
1119           > 133
1120           L5D. |   -.1892527   .0903206    -2.10   0.038   -.3682468
1121           -.0102
1122           > 585
1123           L7D. |    .0639647   .0913902     0.70   0.485   -.1171492
1124           .2450
1125           > 786
1126           |
1127           _cons |    .0025724   .0044848     0.57   0.567   -.0063154
1128           .0114
1129           > 602
1130           -----
1131           --
1132           ---

```

```

1118
1119     237 . scalar drop _all
1120
1121     238 . quietly forval w=12(12)84 {
1122
1123         239 . scalar list
1124             RWmaxobs84 =          84
1125             RWminobs84 =           2
1126             RWrmse84 = .05259823
1127             RWmaxobs72 =          72
1128             RWminobs72 =           2
1129             RWrmse72 = .05283772
1130             RWmaxobs60 =          60
1131             RWminobs60 =           2
1132             RWrmse60 = .05314168
1133             RWmaxobs48 =          48
1134             RWminobs48 =           2
1135             RWrmse48 = .0530381
1136             RWmaxobs36 =          36
1137             RWminobs36 =           2
1138             RWrmse36 = .05353125
1139             RWmaxobs24 =          24
1140             RWminobs24 =           2
1141             RWrmse24 = .05282122
1142             RWmaxobs12 =          12
1143             RWminobs12 =           2
1144             RWrmse12 = .06036464
1145
1146         240 . /*
1147             > RWmaxobs84 =          84
1148             > RWminobs84 =           2
1149             > RWrmse84 = .05250414
1150             > */
1151         241 .
1152         242 . drop pflcount ub1 ub2 ub3 lb1 lb2 lb3
1153
1154         243 .
1155         244 . scalar rwrms = .05250414
1156
1157         245 . reg d.lnWeeklyEarnings l(3,5,7)d.lnWeeklyEarnings if tin(,2021m2)
1158
1159             Source |          SS          df          MS      Number of obs      =
1160             -----+-----
1161             3.85
1162             Model |    .026396014          3    .008798671      Prob > F          =
1163             0.0115
1164             Residual |    .251283445        110    .002284395      R-squared          =
1165             0.0951
1166             -----+-----
1167             0.0704
1168             Adj R-squared          =

```

```

1164      Total | .277679459      113      .00245734      Root MSE      =
      .0478
1165
1166      -----
      --
1167      ---
1168      D. |
1169      lnWeeklyEarnings |      Coef.      Std. Err.      t      P>|t|      [95% Conf.
      Interv
1170      > a1]
1171      -----+-----
      --
1172      ---
1173      lnWeeklyEarnings |
1174      L3D. |      -.2408947      .0906673      -2.66      0.009      -.4205761
      -.0612
1175      > 133
1176      L5D. |      -.1892527      .0903206      -2.10      0.038      -.3682468
      -.0102
1177      > 585
1178      L7D. |      .0639647      .0913902      0.70      0.485      -.1171492
      .2450
1179      > 786
1180      |
1181      _cons |      .0025724      .0044848      0.57      0.567      -.0063154
      .0114
1182      > 602
1183      -----
      --
1184      ---
1185
1186      246 . predict pd
1187      (option xb assumed; fitted values)
1188      (260 missing values generated)
1189
1190      247 . gen pflcount=exp((rwrmsc^2)/2)*exp(1.lnWeeklyEarnings+pd) if
      Date==tm(2021m3)
1191      (374 missing values generated)
1192
1193      248 . gen ub1=exp((rwrmsc^2)/2)*exp(1.lnWeeklyEarnings+pd+1*rwrmsc) if
      Date==tm(202
1194      > 1m3)
1195      (374 missing values generated)
1196
1197      249 . gen lb1=exp((rwrmsc^2)/2)*exp(1.lnWeeklyEarnings+pd-1*rwrmsc) if
      Date==tm(202
1198      > 1m3)
1199      (374 missing values generated)
1200
1201      250 . gen ub2=exp((rwrmsc^2)/2)*exp(1.lnWeeklyEarnings+pd+2*rwrmsc) if
      Date==tm(202

```

```

1202         > lm3)
1203         (374 missing values generated)
1204
1205     251 . gen lb2=exp((rwrms2)/2)*exp(1.lnWeeklyEarnings+pd-2*rwrms2) if
Date==tm(202
1206         > lm3)
1207         (374 missing values generated)
1208
1209     252 . gen ub3=exp((rwrms2)/2)*exp(1.lnWeeklyEarnings+pd+3*rwrms2) if
Date==tm(202
1210         > lm3)
1211         (374 missing values generated)
1212
1213     253 . gen lb3=exp((rwrms2)/2)*exp(1.lnWeeklyEarnings+pd-3*rwrms2) if
Date==tm(202
1214         > lm3)
1215         (374 missing values generated)
1216
1217     254 . drop pd
1218
1219     255 .
1220     256 . replace pflcount=WeeklyEarnings if Date==tm(2021m2)
1221         (1 real change made)
1222
1223     257 . replace ub1=WeeklyEarnings if Date==tm(2021m2)
1224         (1 real change made)
1225
1226     258 . replace ub2=WeeklyEarnings if Date==tm(2021m2)
1227         (1 real change made)
1228
1229     259 . replace ub3=WeeklyEarnings if Date==tm(2021m2)
1230         (1 real change made)
1231
1232     260 . replace lb1=WeeklyEarnings if Date==tm(2021m2)
1233         (1 real change made)
1234
1235     261 . replace lb2=WeeklyEarnings if Date==tm(2021m2)
1236         (1 real change made)
1237
1238     262 . replace lb3=WeeklyEarnings if Date==tm(2021m2)
1239         (1 real change made)
1240
1241     263 .
1242     264 . twoway (tsrline ub3 ub2 if tin(2020m3,2021m3), ///
1243         > recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
1244         > (tsrline ub2 ub1 if tin(2020m3,2021m3), ///
1245         > recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
1246         > (tsrline ub1 pflcount if tin(2020m3,2021m3), ///
1247         > recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///
1248         > (tsrline pflcount lb1 if tin(2020m3,2021m3), ///
1249         > recast(rarea) fcolor(khaki) fintensity(65) lwidth(none) ) ///

```



```

1250      >      (tsrline lb1 lb2 if tin(2020m3,2021m3), ///
1251      >      recast(rarea) fcolor(khaki) fintensity(40) lwidth(none) ) ///
1252      >      (tsrline lb2 lb3 if tin(2020m3,2021m3), ///
1253      >      recast(rarea) fcolor(khaki) fintensity(20) lwidth(none) ) ///
1254      >      (tsline WeeklyEarnings pflcount if tin(2020m3,2021m3) , ///
1255      >      lcolor(gs12 teal) lwidth(medthick medthick) ///
1256      >      lpattern(solid longdash)) ///
1257      >      (scatter withMarchEarnings Date if tin(2021m3,)), scheme(slmono)
lege
1258      > nd(off)
1259
1260      265 . graph export "WeeklyFan.png", replace
1261      (file /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time Series/Problem
Sets
1262      > /Final Project/WeeklyFan.png written in PNG format)
1263
1264      266 .
1265      267 . log close
1266      name: <unnamed>
1267      log: /Users/guslipkin/Documents/Spring2020/CAP 4763 ~ Time
Series/Probl
1268      > em Sets/Final Project/Final Project.smcl
1269      log type: smcl
1270      closed on: 19 Apr 2021, 21:08:41
1271      -----
--

```