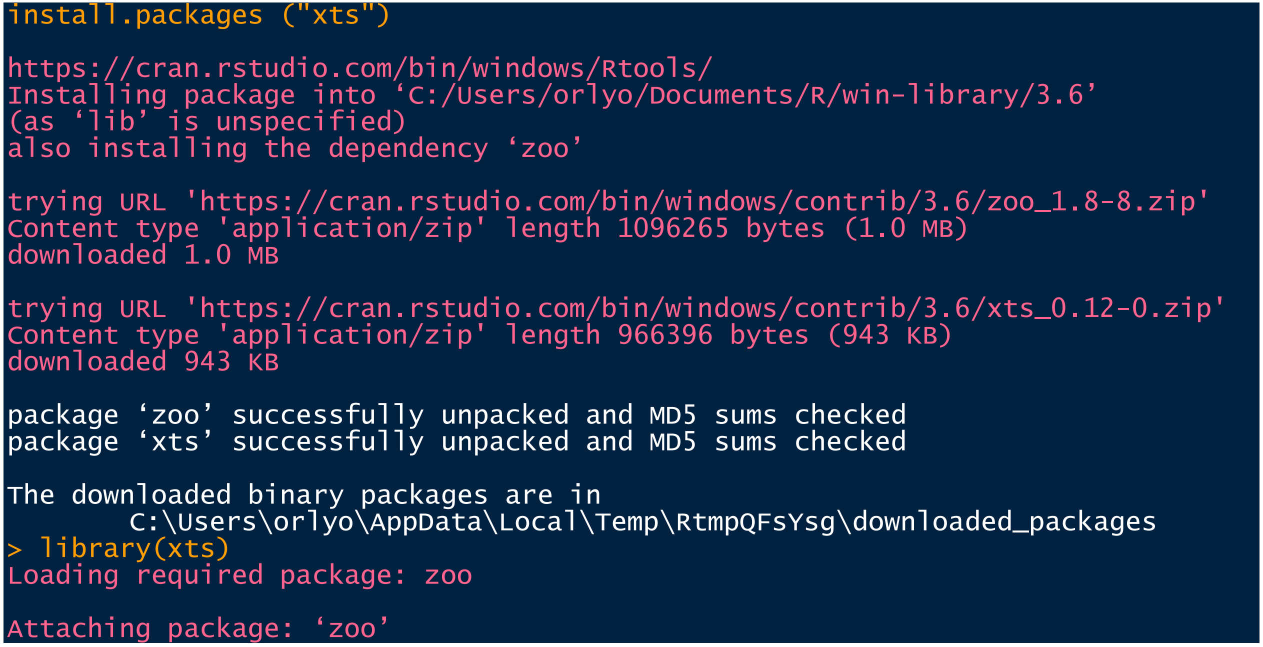
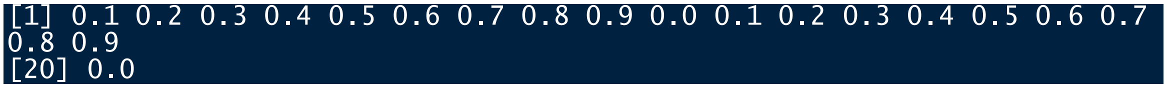
STAT 2321
Homework 1
Orly Olbum
8/27/2020

1. I installed R and RStudio.
2. I installed packages “xts” and “zoo”.



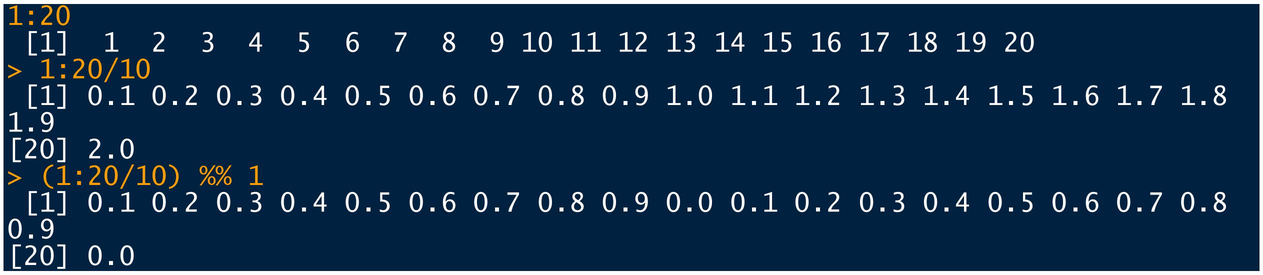
```
install.packages ("xts")

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/orlyo/Documents/R/win-library/3.6’
(as ‘lib’ is unspecified)
also installing the dependency ‘zoo’

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/zoo_1.8-8.zip'
Content type 'application/zip' length 1096265 bytes (1.0 MB)
downloaded 1.0 MB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/xts_0.12-0.zip'
Content type 'application/zip' length 966396 bytes (943 KB)
downloaded 943 KB

package ‘zoo’ successfully unpacked and MD5 sums checked
package ‘xts’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\orlyo\AppData\Local\Temp\RtmpQFsYsg\downloaded_packages
> library(xts)
Loading required package: zoo

Attaching package: ‘zoo’
```

3. Installed and loaded package “astsa” from github.
4. “(1:20/10) %% 1” produces the following output:



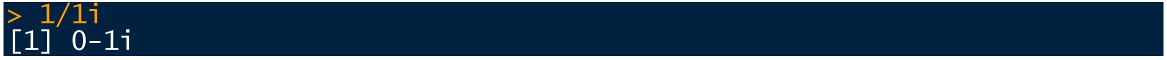
```
[1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
0.8 0.9
[20] 0.0
```

If we were just to look at 1:20, we would get the sequence of numbers from 1 to 20. 1:20/10 returns the sequence, but each number divided by 10. With (1:20/10) %% 1, we get 1:20/10 mod 1, or the remainder.



```
1:20
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
> 1:20/10
 [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8
1.9
[20] 2.0
> (1:20/10) %% 1
 [1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8
0.9
[20] 0.0
```

5. Verify that 1/ i = − i where i = sqrt(− 1) .



```
> 1/1i
[1] 0-1i
```

6. e^ipi – display in R

```
> exp(1)^(1i * pi)
[1] -1+0i
```

7. Calculate cos(pi/2), cos(pi), cos(3pi/2), cos(2pi)

```
> cos(pi*1:4/2)
[1]  6.123032e-17 -1.000000e+00 -1.836910e-16  1.000000e+00
```

8. The difference between two lines, 0 = x = y and 0 -> x -> y: as seen below, the operators have specific requirements for direction, so with “=” the recycling rule doesn’t work, whereas with “->” it does.

```
> 0 = x = y
Error in 0 = x = y : invalid (do_set) left-hand side to assignment
> 0 -> x -> y
> x
[1] 0
> y
[1] 0
```

9. The vector y+z gave a warning (below) because the vectors are different lengths. R recycles the values in z to add to y, but the warning states that the longer vector is not a multiple of the shorter vector, so not all values are recycled.

```
> y + z
[1] 10  7  4
Warning message:
In y + z : longer object length is not a multiple of shorter object length
```

10. I created a directory on my desktop for this class, separate from other classes that are using R. Loaded astsa and looked at cpg (below).

```
> library(astsa)
> help(cpg)
> write(cpg, file = "zzz.txt", ncolumns = 1)
```

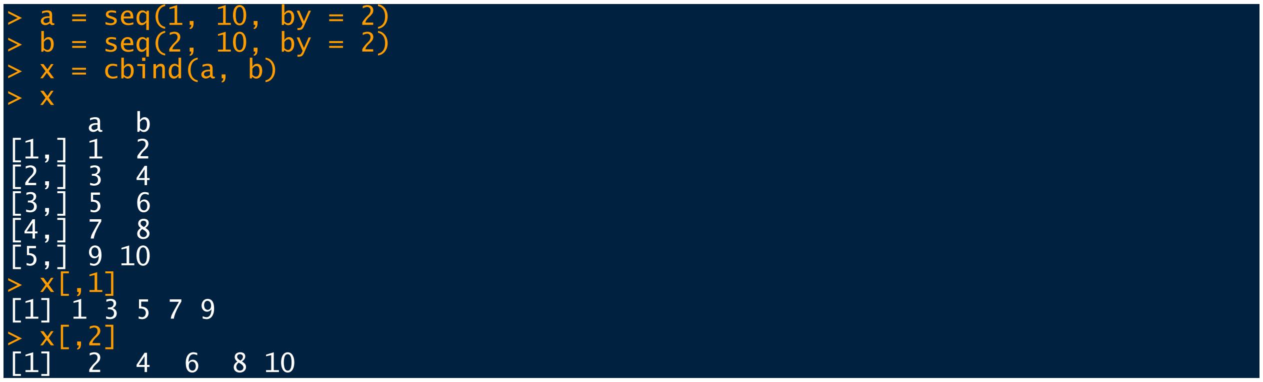
11. Found the txt file created in the working directory.

```
> getwd()
[1] "C:/Desktop/Grad School/Fall 2020/STAT 2321 - Applied Advanced Time Serie
s/Homeworks"
```

12. Scanned and viewed data “zzz” from my working directory.

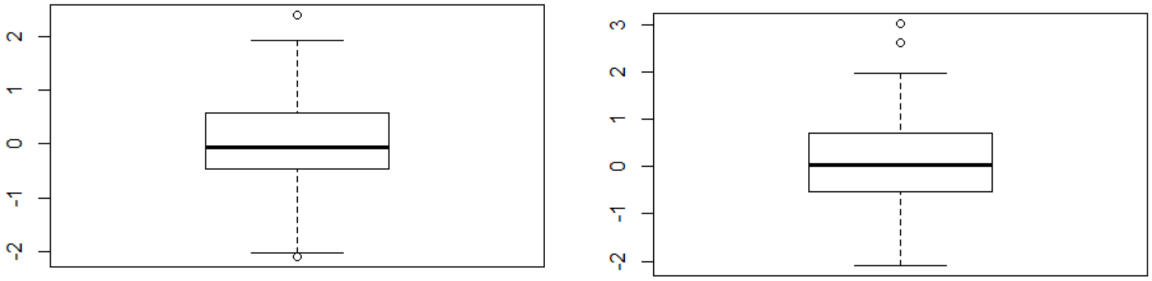
```
> cost_per_gig = scan("zzz.txt")
Read 29 items
> cost_per_gig
 [1] 2.13e+05 2.95e+05 2.60e+05 1.75e+05 1.60e+05
 [6] 7.10e+04 6.00e+04 3.00e+04 3.60e+04 9.00e+03
[11] 7.00e+03 4.00e+03 2.00e+03 9.50e+02 8.65e+02
[16] 2.59e+02 1.03e+02 6.29e+01 2.45e+01 1.25e+01
[21] 6.41e+00 2.68e+00 1.57e+00 1.38e+00 6.70e-01
[26] 5.30e-01 4.20e-01 2.70e-01 7.00e-02
```

13. See R output below:

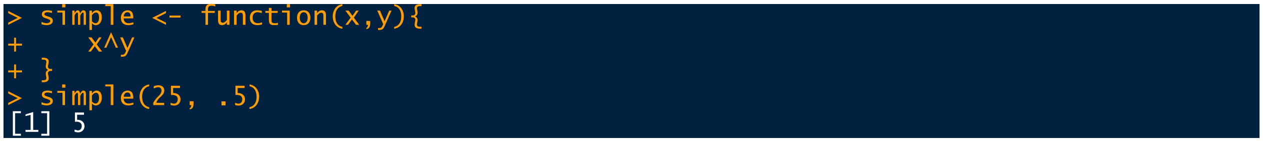


```
> a = seq(1, 10, by = 2)
> b = seq(2, 10, by = 2)
> x = cbind(a, b)
> x
     a  b
[1,] 1  2
[2,] 3  4
[3,] 5  6
[4,] 7  8
[5,] 9 10
> x[,1]
[1] 1 3 5 7 9
> x[,2]
[1]  2  4  6  8 10
```

14. See plots below:

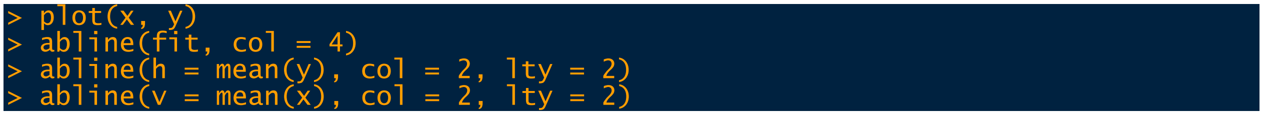


15. See output below:

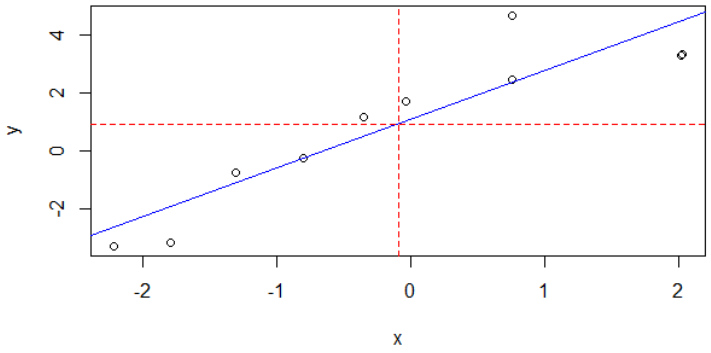


```
> simple <- function(x,y){
+    x^y
+ }
> simple(25, .5)
[1] 5
```

16. See output and plot below:



```
> plot(x, y)
> abline(fit, col = 4)
> abline(h = mean(y), col = 2, lty = 2)
> abline(v = mean(x), col = 2, lty = 2)
```

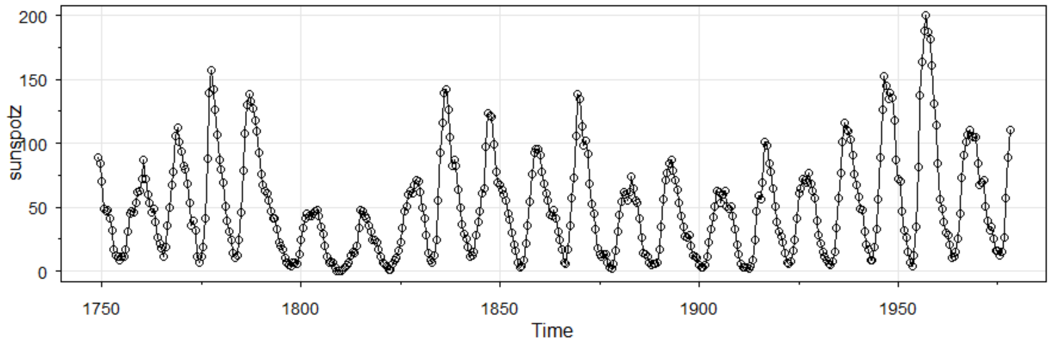


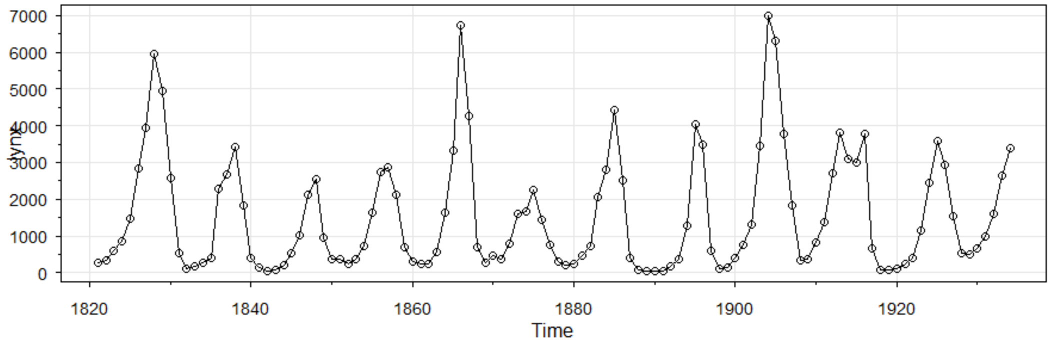
17. See output below (part4 has NAs, has been kicked out of regression):

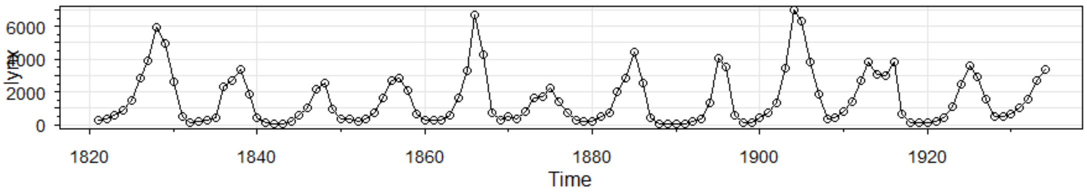
```
> part4 <- lag(part, -4)
> summary(fit <- lm(cmort ~ part + part4, na.action = NULL))

Call:
lm(formula = cmort ~ part + part4, na.action = NULL)

Residuals:
    Min      1Q  Median      3Q     Max 
-21.609  -6.328  -0.626   5.385  36.253 

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 74.79860    1.30943   57.12   <2e-16 ***
part         0.29317    0.02631   11.14   <2e-16 ***
part4             NA         NA      NA       NA    
---
Signif. codes:  
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 8.969 on 506 degrees of freedom
Multiple R-squared:  0.197,	Adjusted R-squared:  0.1954 
F-statistic: 124.2 on 1 and 506 DF,  p-value: < 2.2e-16
```

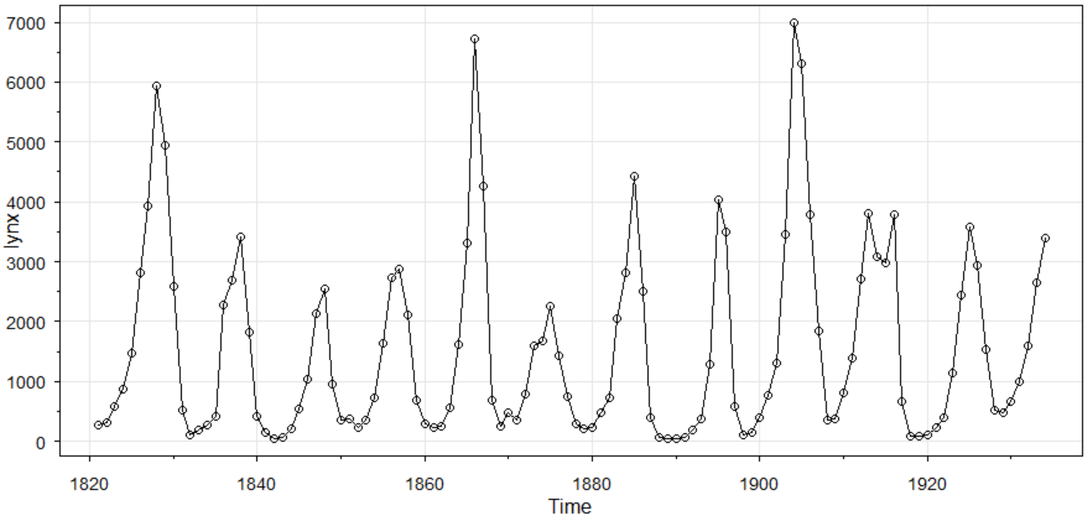
18. See plots below. With a few different views we can see that rather than rising quickly and falling slowly like the sunspotz data, the lynx data rises slowly and falls quickly. A second plot putting them together, similar to the book’s example, gives a better view (second plots).

```
> par(mfrow = c(2, 1))
> tsplot(sunspotz, type = “o”)
> tsplot(lynx, type = “o”)
```


sunspotz
200
150
100
50
0
1750
1800
1850
1900
1950
Time


lynx
7000
6000
5000
4000
3000
2000
1000
0
1820
1840
1860
1880
1900
1920
Time


lynx
6000
4000
2000
0
1820
1840
1860
1880
1900
1920
Time


lynx
7000
6000
5000
4000
3000
2000
1000
0
1820
1840
1860
1880
1900
1920
Time