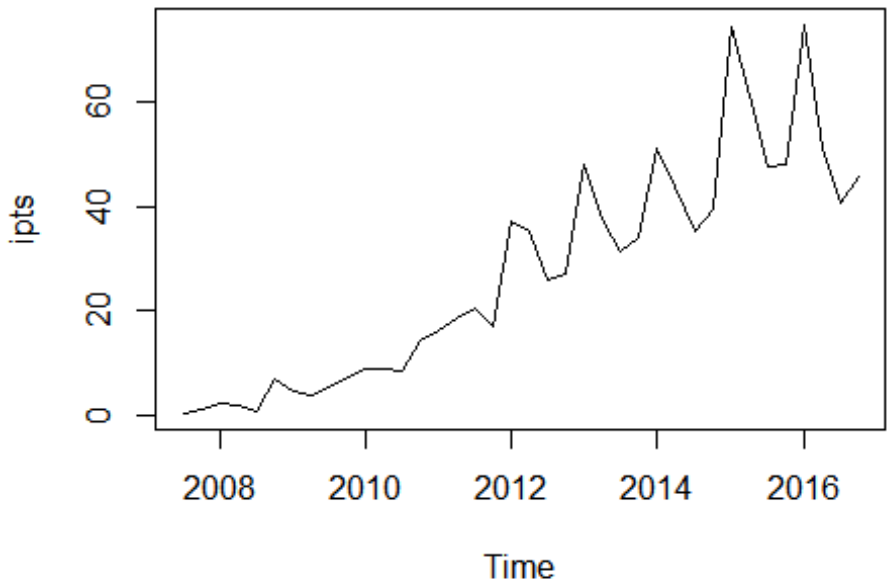
Timeseries.R

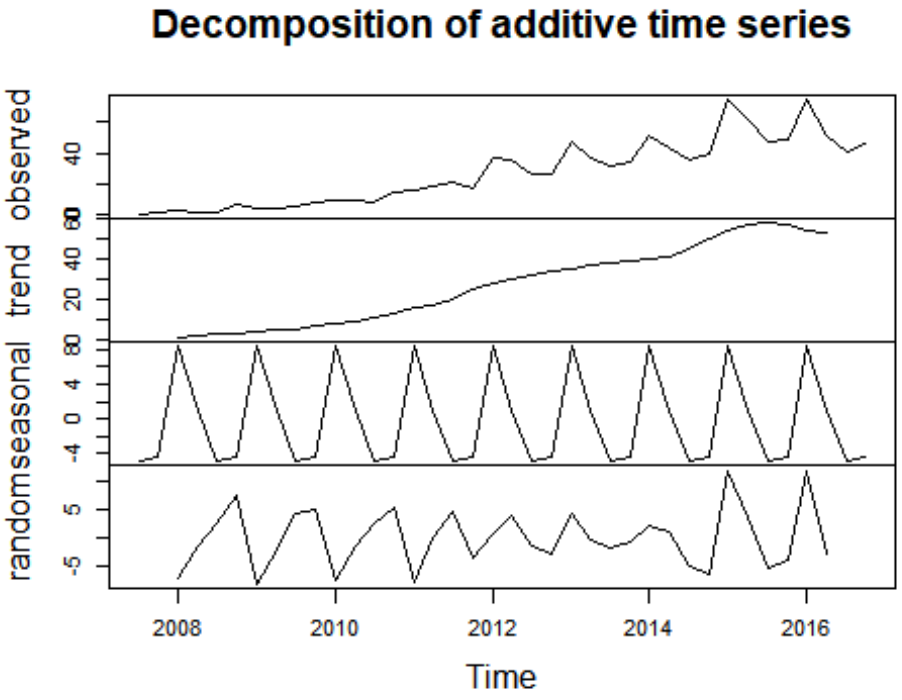
Fiona

Fri Oct 13 15:57:26 2017

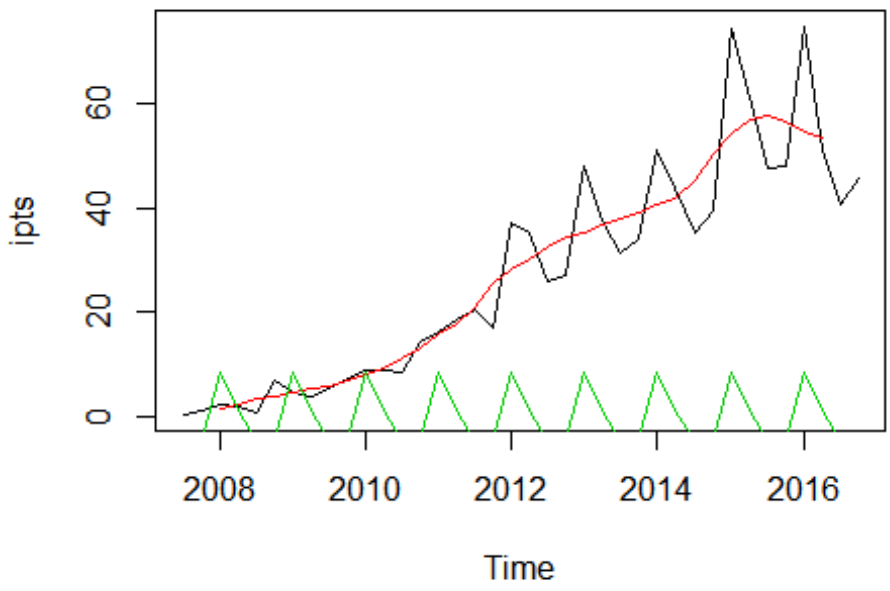
```
# read in data from csv file
data <- read.csv("C:/Users/Fiona/Desktop/Business Analytics and Decision
Sciences/Forecasting and Business Analytics/FABA-L2-Notes/iphonesales.csv",
header=TRUE)

#save as time series data
ipts <- ts(data$Sales,start=c(2007, 3), end=c(2016, 4), frequency=4)

# plot the time series data
plot(ipts)
```



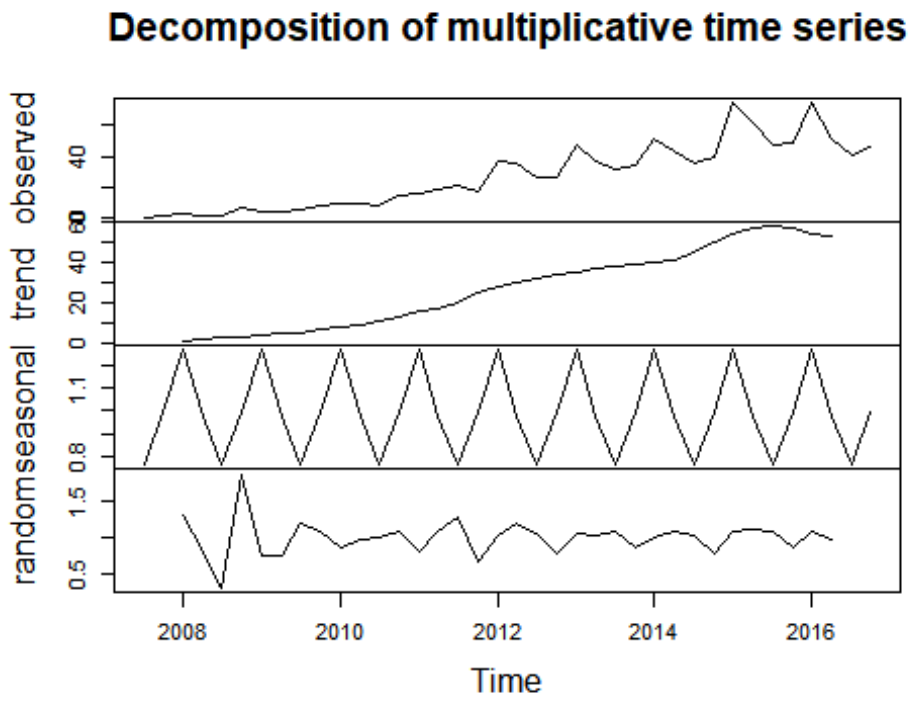

```
# Use additive decomposition
de <- decompose(ipts, type="additive")

# Plot the decomposition
plot(de)
```



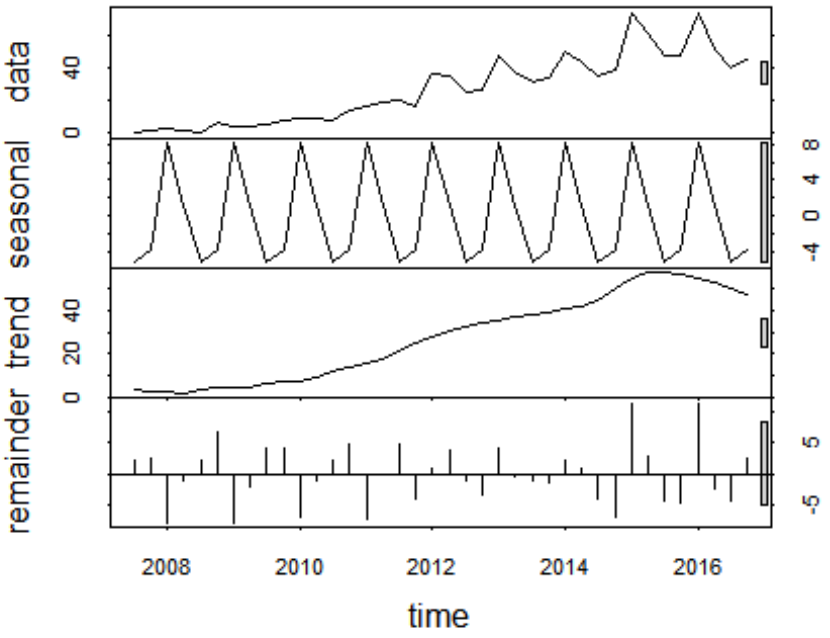

```r
# plot data with trend and seasonal components
plot(ipts)
lines(de$trend, col=2)
lines(de$seasonal, col=3)
```



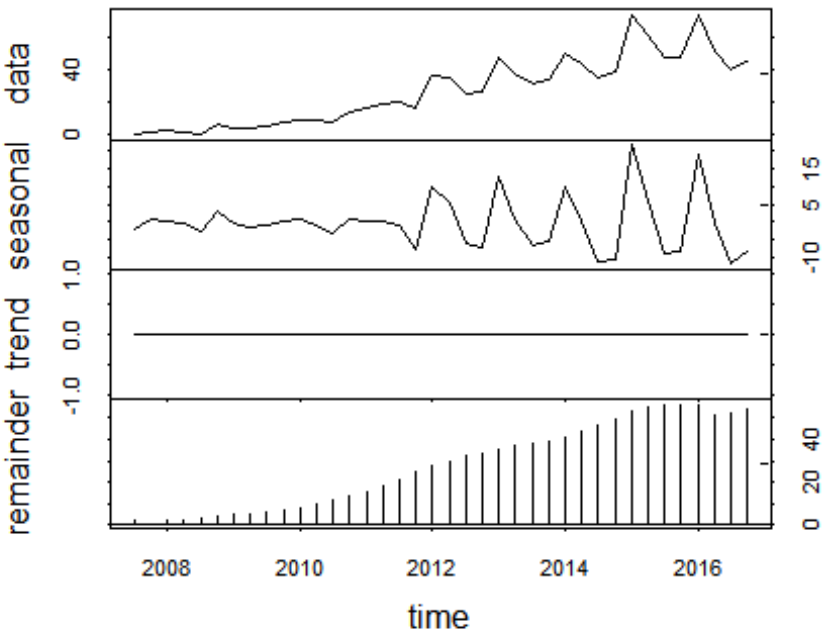

```
# using multiplicative
de <- decompose(ipts, type="multiplicative")
plot(de)
```



```r
# Loess Decomposition
lo <- stl(ipts, s.window="periodic")
plot(lo)
```




```r
lo <- stl(ipts, s.window=1)
plot(lo)
```




```
# log of ts data for multiplicative
lipts <- log(ipts)
lo <- stl(lipts, s.window="periodic")
lo$time.series
```

```
##            seasonal      trend   remainder
## 2007 Q3 -0.33090899 -0.5874225 -0.39100187
## 2007 Q4  0.02356623 -0.2101199  0.29988231
## 2008 Q1  0.29367049  0.1497006  0.39819611
## 2008 Q2  0.01367134  0.4892535  0.02770339
## 2008 Q3 -0.33090899  0.7947947 -0.79238973
## 2008 Q4  0.02356623  1.0339054  0.87259946
## 2009 Q1  0.29367049  1.3638485 -0.18504689
## 2009 Q2  0.01367134  1.5771230 -0.25842834
## 2009 Q3 -0.33090899  1.7185575  0.26293134
## 2009 Q4  0.02356623  1.9028820  0.07096945
## 2010 Q1  0.29367049  2.0411110 -0.16687133
## 2010 Q2  0.01367134  2.1897657 -0.03438330
## 2010 Q3 -0.33090899  2.3681193  0.09102137
## 2010 Q4  0.02356623  2.5298990  0.09270952
## 2011 Q1  0.29367049  2.7156390 -0.22183218
## 2011 Q2  0.01367134  2.8766768  0.03549802
## 2011 Q3 -0.33090899  3.0095617  0.33393665
## 2011 Q4  0.02356623  3.1617892 -0.34803290
## 2012 Q1  0.29367049  3.2763812  0.04194672
## 2012 Q2  0.01367134  3.3972141  0.14617541
## 2012 Q3 -0.33090899  3.4669099  0.12324884
```

```
## 2012 Q4  0.02356623  3.4907518 -0.22182007
## 2013 Q1  0.29367049  3.5259502  0.04719575
## 2013 Q2  0.01367134  3.5964547  0.01234645
## 2013 Q3 -0.33090899  3.6265706  0.14603767
## 2013 Q4  0.02356623  3.6418869 -0.14499232
## 2014 Q1  0.29367049  3.6732522 -0.03450902
## 2014 Q2  0.01367134  3.7262875  0.03784678
## 2014 Q3 -0.33090899  3.7835024  0.10845264
## 2014 Q4  0.02356623  3.8616082 -0.21471361
## 2015 Q1  0.29367049  3.9483333  0.06839261
## 2015 Q2  0.01367134  4.0318201  0.06816549
## 2015 Q3 -0.33090899  4.0458045  0.14646555
## 2015 Q4  0.02356623  4.0133879 -0.16471198
## 2016 Q1  0.29367049  3.9693803  0.05149967
## 2016 Q2  0.01367134  3.9447848 -0.02291190
## 2016 Q3 -0.33090899  3.9112995  0.11843925
## 2016 Q4  0.02356623  3.8775274 -0.08316157
```

```
newlo = exp(lo$time.series)
newlo
```

```
##          seasonal      trend remainder
## 2007 Q3 0.7182705  0.5557579 0.6763789
## 2007 Q4 1.0238461  0.8104871 1.3497000
## 2008 Q1 1.3413418  1.1614864 1.4891360
## 2008 Q2 1.0137652  1.6310982 1.0280907
## 2008 Q3 0.7182705  2.2139863 0.4527615
## 2008 Q4 1.0238461  2.8120265 2.3931236
## 2009 Q1 1.3413418  3.9112165 0.8310653
## 2009 Q2 1.0137652  4.8410083 0.7722644
## 2009 Q3 0.7182705  5.5764786 1.3007374
## 2009 Q4 1.0238461  6.7051911 1.0735484
## 2010 Q1 1.3413418  7.6991584 0.8463085
## 2010 Q2 1.0137652  8.9331195 0.9662011
## 2010 Q3 0.7182705 10.6772928 1.0952924
## 2010 Q4 1.0238461 12.5522389 1.0971430
## 2011 Q1 1.3413418 15.1142653 0.8010498
## 2011 Q2 1.0137652 17.7551708 1.0361356
## 2011 Q3 0.7182705 20.2785105 1.3964547
## 2011 Q4 1.0238461 23.6128065 0.7060756
## 2012 Q1 1.3413418 26.4797741 1.0428389
## 2012 Q2 1.0137652 29.8807403 1.1573992
## 2012 Q3 0.7182705 32.0375889 1.1311659
## 2012 Q4 1.0238461 32.8106056 0.8010595
## 2013 Q1 1.3413418 33.9860507 1.0483272
## 2013 Q2 1.0137652 36.4687137 1.0124230
## 2013 Q3 0.7182705 37.5837074 1.1572398
## 2013 Q4 1.0238461 38.1637800 0.8650289
## 2014 Q1 1.3413418 39.3797695 0.9660796
## 2014 Q2 1.0137652 41.5246632 1.0385721
```

```
## 2014 Q3 0.7182705 43.9697735 1.1145521
## 2014 Q4 1.0238461 47.5417489 0.8067725
## 2015 Q1 1.3413418 51.8488758 1.0707856
## 2015 Q2 1.0137652 56.3634022 1.0705425
## 2015 Q3 0.7182705 57.1571524 1.1577350
## 2015 Q4 1.0238461 55.3340187 0.8481380
## 2016 Q1 1.3413418 52.9517068 1.0528488
## 2016 Q2 1.0137652 51.6652168 0.9773486
## 2016 Q3 0.7182705 49.9638392 1.1257385
## 2016 Q4 1.0238461 48.3046302 0.9202025
```

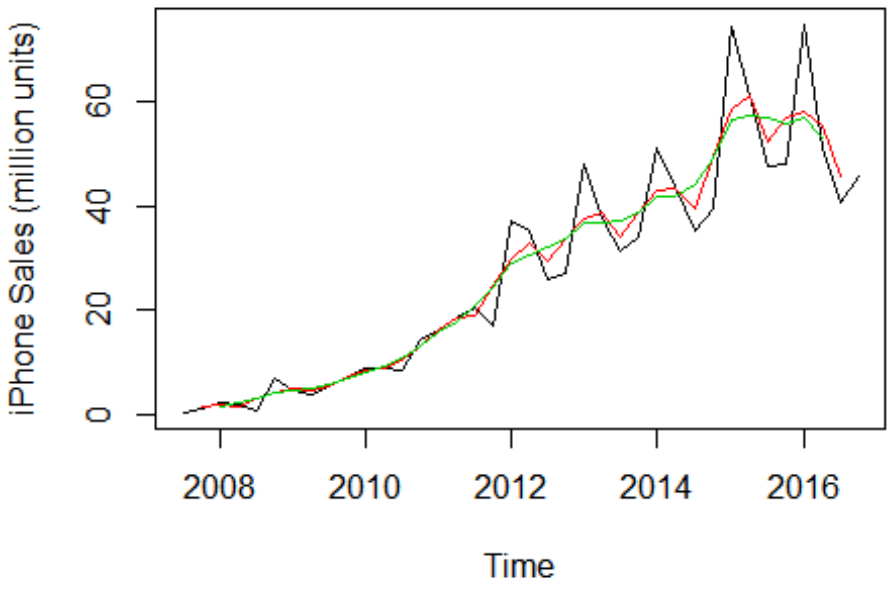
```
#install.packages("forecast")
library("forecast")
```

```
## Warning: package 'forecast' was built under R version 3.3.3
```

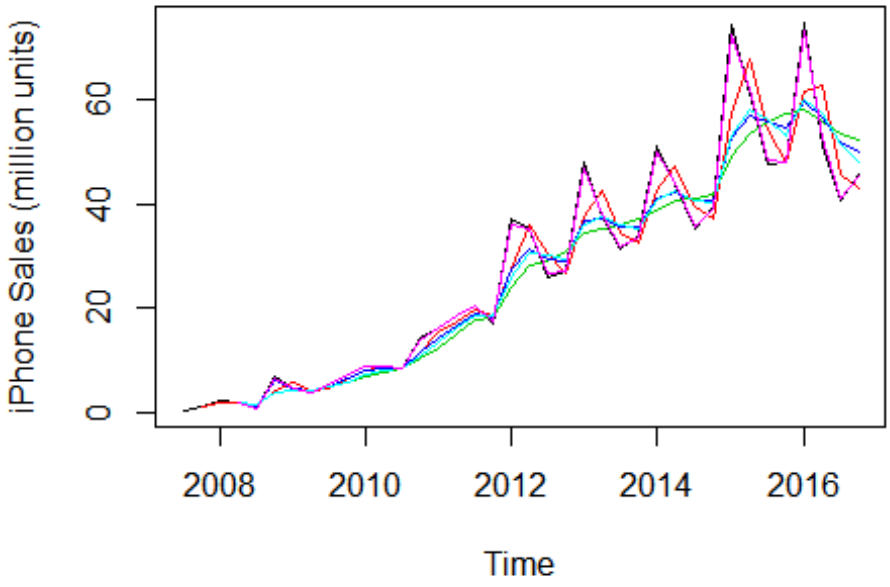
```
plot(ipts, ylab="iPhone Sales (million units)")

# Simple Moving Average with n = 3
sma <- ma(ipts, order=3)

# Check calculation of the first value
mean(ipts[1:3])
```

```
## [1] 1.236667
```

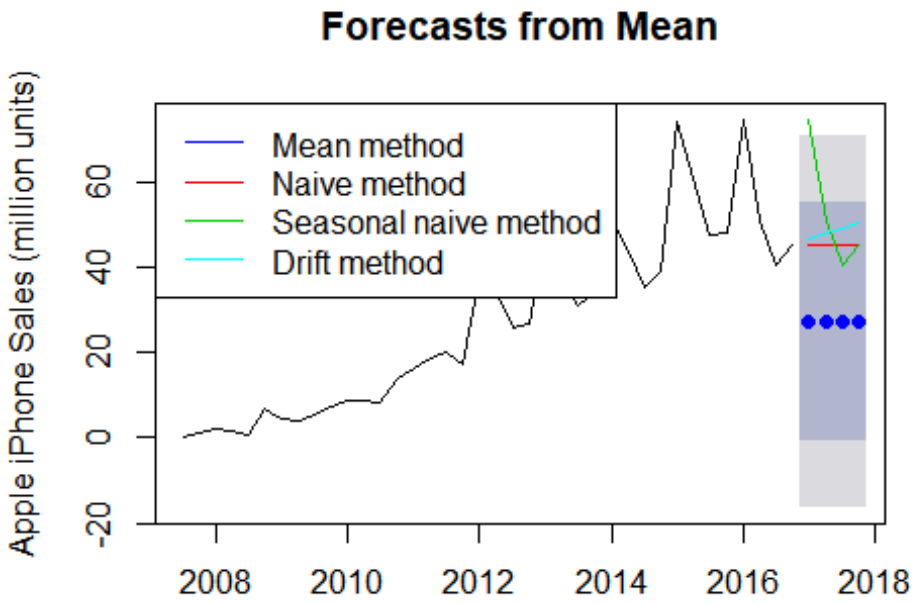
```
lines(sma, col=2)

# Double Moving Average (use this to smooth the data further!)
sma2 <- ma(sma, order=3)
lines(sma2, col=3)
```




```
# WMA is not supported in the forecast package so will use TTR package
#install.packages("TTR")
library("TTR")

## Warning: package 'TTR' was built under R version 3.3.3

plot(ipts, ylab="iPhone Sales (million units)")

# WMA is similar to an EMA, but with linear weighting if the length of w is
equal to n. If the length of w is equal to the length of x, the WMA will use
the values of w as weights.
# Weighted Moving Average with n = 4 and w for t
wma <- WMA(ipts, n=2, w = 1:38)
lines(wma, col="red")
wma <- WMA(ipts, n=4, w=(1:38)^2)
lines(wma, col=3)
wma <- WMA(ipts, n=4, w=(1:38)^10)
lines(wma, col=4)

# Weighted Moving Average with n = 4 and w for n
wma <- WMA(ipts, n=4, w=1:4)
lines(wma, col=5)
wma <- WMA(ipts, n=4, w=(1:4)^10)
lines(wma, col=6)
```




```
# average method
ifit1 <- meanf(ipts, h=4)
plot(ifit1, ylab="Apple iPhone Sales (million units)")

# naïve method
ifit2 <- naive(ipts, h=4)
lines(ifit2$mean,col=2)

# seasonal naïve method
ifit3 <- snaive(ipts, h=4)
lines(ifit3$mean,col=3)

# drift method
ifit4 <- rwf(ipts,drift=TRUE,h=4)
lines(ifit4$mean,col=5)

# add legend
legend("topleft",lty=1,col=c(4,2,3,5),legend=c("Mean method","Naive
method","Seasonal naive method", "Drift method"))
```


Forecasts from Mean
Apple iPhone Sales (million units)
-20
0
20
40
60
2008
2010
2012
2014
2016
2018
Mean method
Naive method
Seasonal naive method
Drift method