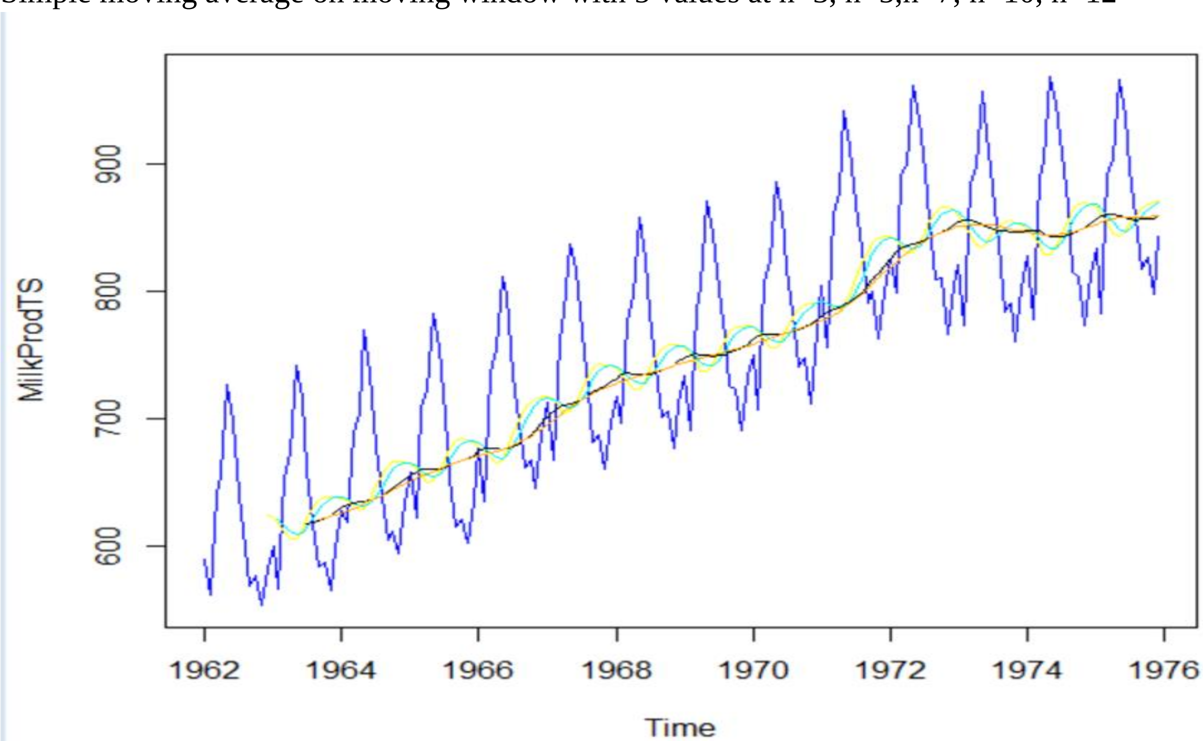
Q1 Summary:

```
> # Confidence Interval
>
> mean(Accy_svm) - Accy_svm_err;
[1] 0.6542986
> mean(Accy_svm) + Accy_svm_err
[1] 0.6796488
>
> summary(Accy_svm)
   Min.  1st Qu.  Median    Mean  3rd Qu.    Max.
 0.4605  0.6316  0.6711  0.6670  0.7105  0.8026
```

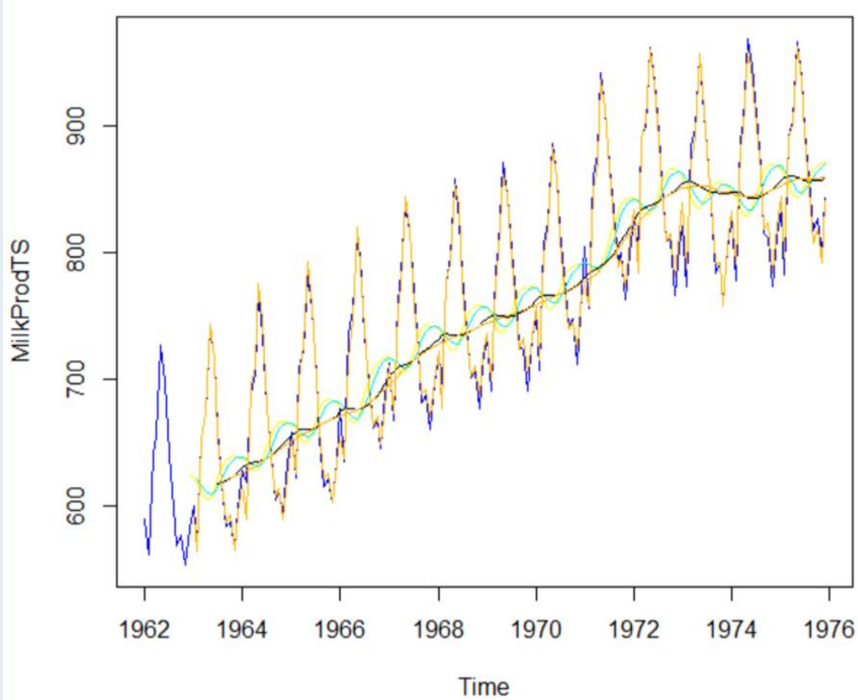
1. SVN fit confidence interval is 65% - 68%
2. We observe that the accuracy of model is higher than LDA, QDA, KNN model.
3. Support Vector machine(SVM) works better than Linear Regression when classes are separable. So, the svm model is good fit.
4. Confidence interval range is higher for SVM model than LDA, QDA, KNN model.

Q2 Summary:

1. Simple moving average on moving window with 5 values at n=3, n=5,n=7, n=10, n=12



2. Exponential plot on simple moving average.



MilkProdTSLog ="blue"

SMATSLogS3="Yellow"

SMATSLogS5 ="Red")

SMATSLogS7 ="cyan2")

SMATSLogS10 ="Black")

SMATSLogS12 ="darkmagenta"

sHWLog$fitted = "darkgoldenrod1

3. Mean forecast error(MFE) and mean absolute deviation on the simple moving average and exponential moving average of n=3.

MFE - SMA	8.519532909
MAD - SMA	59.24989384
MFE - LogSMA	0.011845943
MAD - LogSMA	0.077695424

- From above the Mean forecast error is high => worse performance, the forecast is biased.
- Mean absolute deviation is higher so it’s worse performance.
- Because of low MAD and MFE for Exponential moving average is better than simple moving average.

Question 1:

3_1 [30 points]. This assignment extends from Assignment-2. Q-1, which is reproduced below. Your assignment is to extend Part-b and use SVM, and provide a comparison with a discussion (note - need to do it only for part-b).

Approach:

1. **Reading data**

```
> setwd("C:/Users/putha/Desktop/ISL/Assignment2")
> library(caret)
> library(MASS)
> library(e1071)
> KCW= read.csv("kc_weather_srt.csv",header=T)
> head(KCW)
        Date Temp.F Dew_Point.F Humidity.percentage Sea_Level_Press.in
1  2014-1-1     26          12                  73              30.19
2  2014-1-4     31          18                  68              29.95
3  2014-1-5     10           1                  63              30.24
4 2014-1-10     38          35                  90              29.70
5 2014-1-11     40          30                  75              29.80
6 2014-1-12     49          29                  51              29.64
  Visibility.mi Wind.mph Precip.in Events
1             5        9      0.03   Snow
2             7       11      0.01   Snow
3             5       14      0.02   Snow
4             6        5      0.00   Rain
5             9        7      0.00   Rain
6            10       10      0.00   Rain
```

2. **SVM fit on linear kernel**

```
> nKCW = 366
> ntrain_KCW = 290
> nleft = nKCW - ntrain_KCW
> rep = 100
>
> # Support Vector Mechine
> Accy_svm = dim(rep)
> Prec_svm_Rain = dim(rep)
> Recall_svm_Rain = dim(rep)
> Prec_svm_Snow = dim(rep)
> Recall_svm_Snow = dim(rep)
> Prec_svm_Rain_Thunderstorm = dim(rep)
> Recall_svm_Rain_Thunderstorm = dim(rep)
>
> for (k in 1:rep) {
+ train = sample(1: nKCW, ntrain_KCW)
+ KCW.svm = svm(Events~.,data=KCW[train,])
+ KCW.svm_predict = predict(KCW.svm,KCW[-train,], type="response", kernal="linear")
+ TabIn = table(KCW[-train,]$Events,KCW.svm_predict)
+ Accy_svm[k] = sum(diag(TabIn))/nleft
+ Prec_svm_Rain[k] = TabIn[1,1]/(TabIn[1,1]+TabIn[2,1]+TabIn[3,1])
+ Recall_svm_Rain[k] = TabIn[1,1]/(TabIn[1,1]+TabIn[1,2]+TabIn[1,3])
+ Prec_svm_Snow[k] = TabIn[2,2]/(TabIn[1,2]+TabIn[2,2]+TabIn[3,2])
+ Recall_svm_Snow[k] = TabIn[2,2]/(TabIn[2,1]+TabIn[2,2]+TabIn[2,3])
+ Prec_svm_Rain_Thunderstorm[k] = TabIn[3,3]/(TabIn[1,3]+TabIn[2,3]+TabIn[3,3])
+ Recall_svm_Rain_Thunderstorm[k] = TabIn[3,3]/(TabIn[3,1]+TabIn[3,2]+TabIn[3,3])
+ }
>
```

3. **Printing the Table and Summary.**

```
> TabIn
                     KCW.svm_predict
                      Rain Rain_Thunderstorm Snow
  Rain                  28                 7    0
  Rain_Thunderstorm     10                17    0
  Snow                  13                 0    1
> summary(KCW.svm)

Call:
svm(formula = Events ~ ., data = KCW[train, ])


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.002680965

Number of Support Vectors:  278

 ( 135 107 36 )


Number of Classes:  3

Levels:
 Rain Rain_Thunderstorm Snow
```

4. **Printing the accuracy, precision and recall.**

```
> # printing the accuracy, precision and recall
> mean(Accy_svm)
[1] 0.6669737
> mean(Prec_svm_Rain)
[1] 0.6156827
> mean(Recall_svm_Rain)
[1] 0.8458514
> mean(Prec_svm_Snow)
[1] NaN
> mean(Recall_svm_Snow)
[1] 0.3281448
> mean(Prec_svm_Rain_Thunderstorm)
[1] 0.7727313
> mean(Recall_svm_Rain_Thunderstorm)
[1] 0.5940357
```

5. **Standard Deviation of Events.**

```
> # printing the standard deviation
> sd(Accy_svm)
[1] 0.06387964
> sd(Prec_svm_Rain)
[1] 0.08390756
> sd(Recall_svm_Rain)
[1] 0.1006516
> sd(Prec_svm_Snow)
[1] NA
> sd(Recall_svm_Snow)
[1] 0.1665864
> sd(Prec_svm_Rain_Thunderstorm)
[1] 0.1087361
> sd(Recall_svm_Rain_Thunderstorm)
[1] 0.1569093
```

6. **Checking the Confidence Interval.**

```
> # Compute 95% confidence interval on Accuracy using t-distribution
> Accy_svm_err = qt(0.975, df = rep-1) * sd(Accy_svm) / sqrt(rep)
>
> # Confidence Interval
>
> mean(Accy_svm) - Accy_svm_err;
[1] 0.6542986
> mean(Accy_svm) + Accy_svm_err
[1] 0.6796488
>
> summary(Accy_svm)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.4605  0.6316  0.6711  0.6670  0.7105  0.8026 
>
```

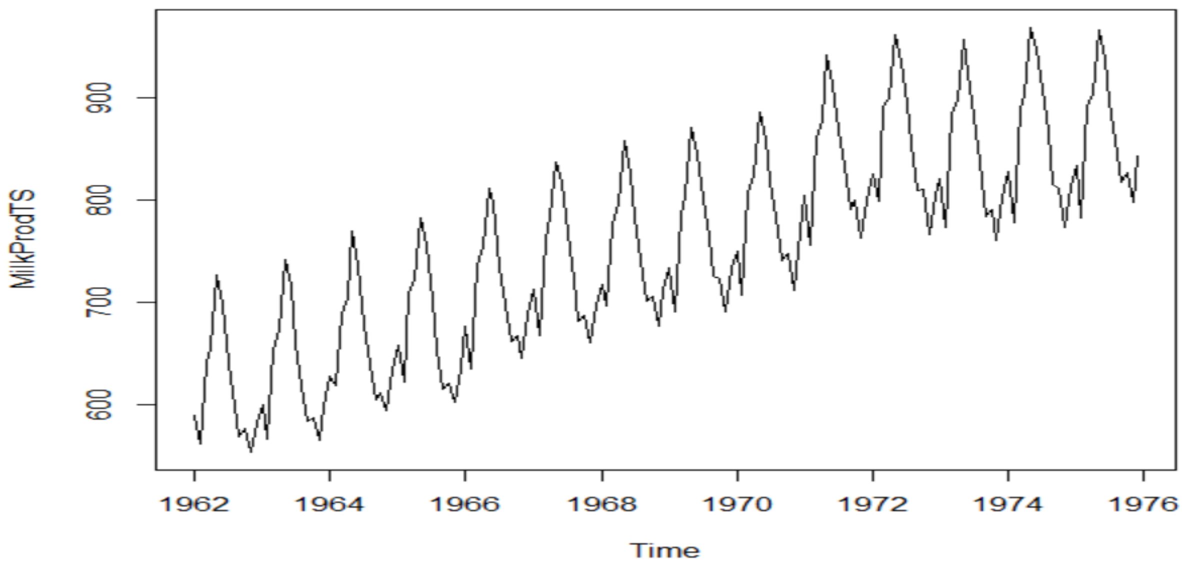
Question 2:

3_2. [45 points] Consider the time series on Milk production data milk-production(1).csv
it shows cow milk production per pound from 1962 to 1975.

a. Try at least three different values for window size with simple moving average (SMA) for forecasting.

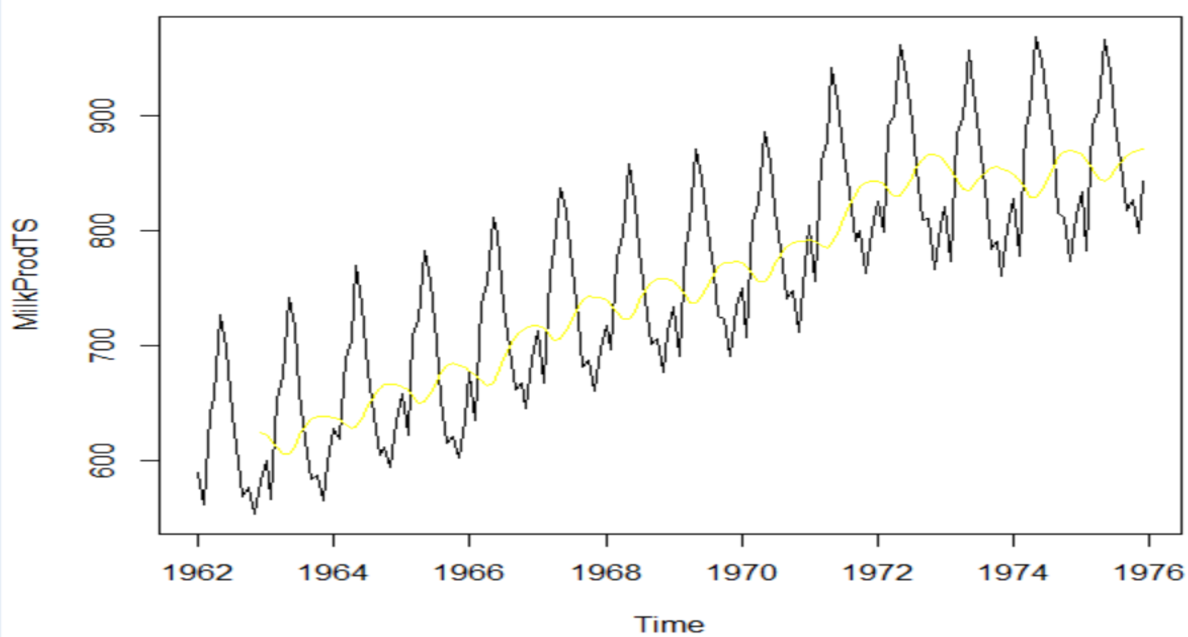
Simple Moving Average(SMA) is a method of time series smoothing and is actually a very basic forecasting technique. It does not need estimation of parameters, but rather is based on order selection.

```
> setwd("C:/Users/putha/Desktop/ISL/Assignment3")
> library(TTR)
Warning message:
package 'TTR' was built under R version 3.4.2
> milkData=read.csv("milk-production(1).csv")
> milkProd<-milkData[,2]
> MilkProdTS<-ts(milkProd,frequency=12,start=c(1962,1))
> MilkProdTS
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1962 589 561 640 656 727 697 640 599 568 577 553 582
1963 600 566 653 673 742 716 660 617 583 587 565 598
1964 628 618 688 705 770 736 678 639 604 611 594 634
1965 658 622 709 722 782 756 702 653 615 621 602 635
1966 677 635 736 755 811 798 735 697 661 667 645 688
1967 713 667 762 784 837 817 767 722 681 687 660 698
1968 717 696 775 796 858 826 783 740 701 706 677 711
1969 734 690 785 805 871 845 801 764 725 723 690 734
1970 750 707 807 824 886 859 819 783 740 747 711 751
1971 804 756 860 878 942 913 869 834 790 800 763 800
1972 826 799 890 900 961 935 894 855 809 810 766 805
1973 821 773 883 898 957 924 881 837 784 791 760 802
1974 828 778 889 902 969 947 908 867 815 812 773 813
1975 834 782 892 903 966 937 896 858 817 827 797 843
> plot.ts(MilkProdTS)
```

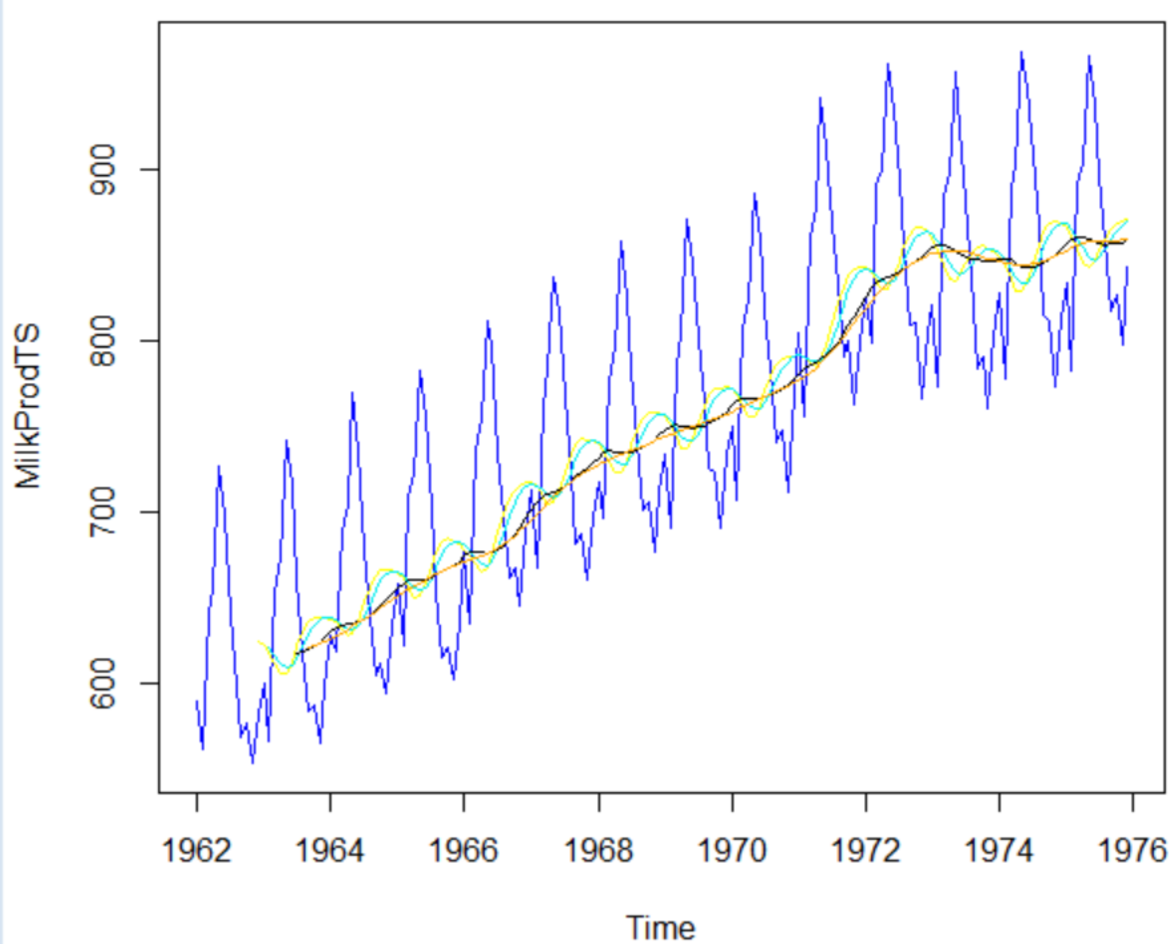


Applying "simple moving average".

```
> plot.ts(MilkProdTS)
> plot(MilkProdTS)
> SMATS3<-SMA(SMA(MilkProdTS,n=3))
> lines(SMATS3,col="Yellow")
```

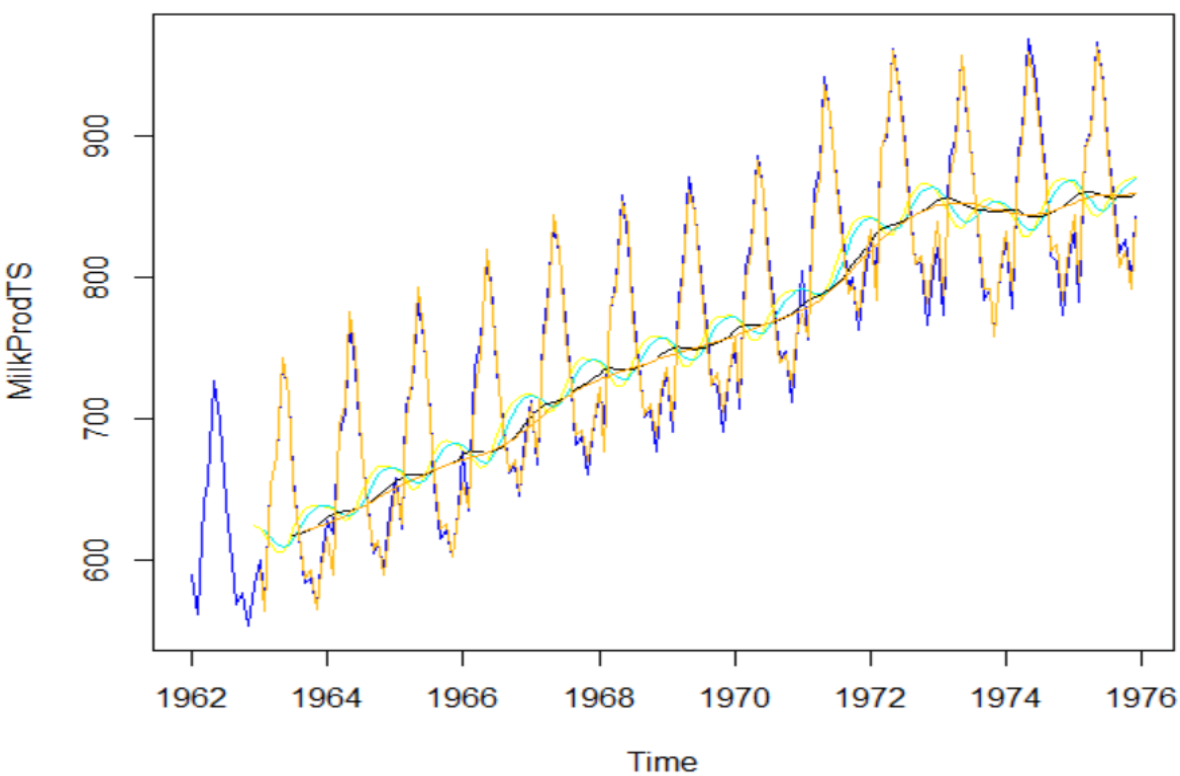


```
> SMATS3<-SMA(SMA(MilkProdTS,n=3))
> SMATS5<-SMA(SMA(MilkProdTS,n=5))
> SMATS7<-SMA(SMA(MilkProdTS,n=7))
> SMATS10<-SMA(SMA(MilkProdTS,n=10))
> SMATS12<-SMA(SMA(MilkProdTS,n=12))
> lines(MilkProdTS,col="blue")
> lines(SMATS3,col="Yellow")
> lines(SMATS5,col="Red")
> lines(SMATS5,col="cyan2")
> lines(SMATS10,col="Black")
> lines(SMATS12,col="Orange")
```

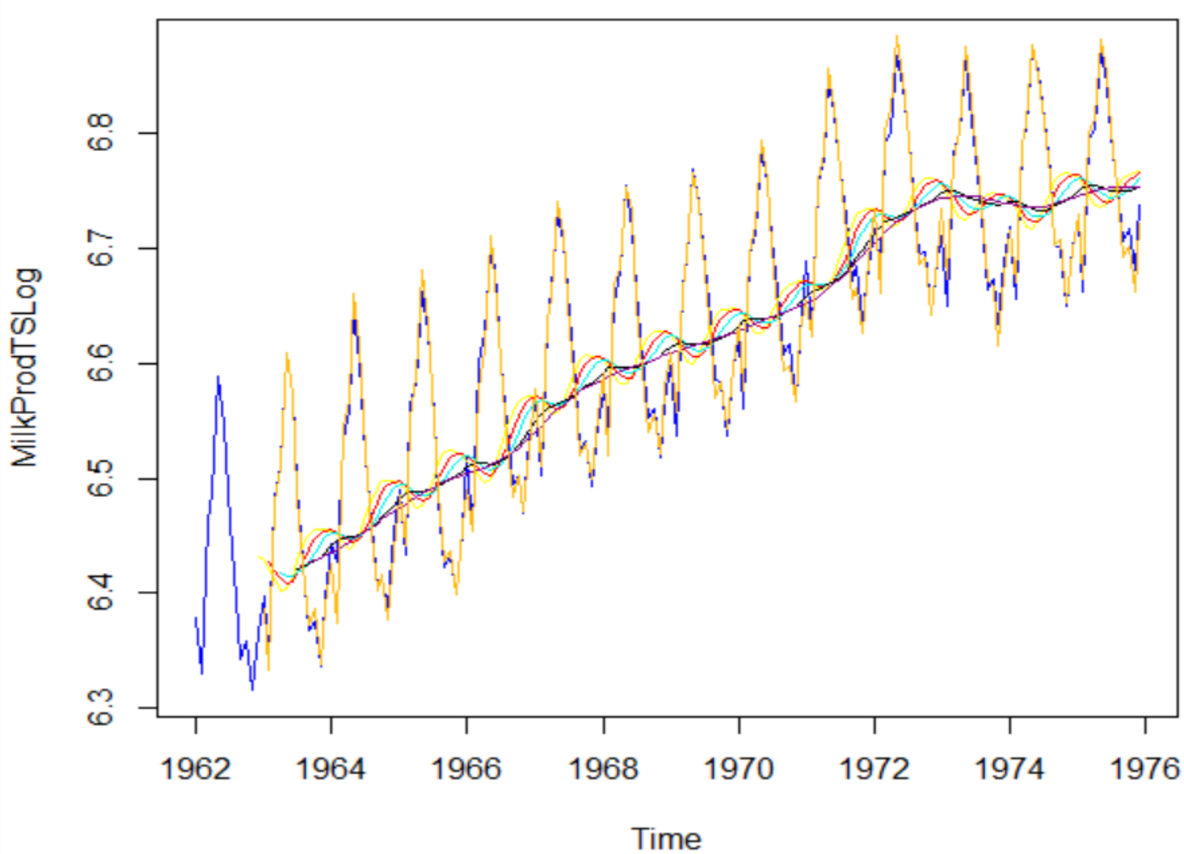


b. Apply exponential moving average using HoltWinters for forecasting.

```
> sHW<-HoltWinters(MilkProdTS)
> sHW
Holt-Winters exponential smoothing with trend and additive seasonal com

Call:
HoltWinters(x = MilkProdTS)

Smoothing parameters:
 alpha: 0.68933
 beta : 0
 gamma: 0.8362592

Coefficients:
          [,1]
a   885.775547
b     1.278118
s1  -16.743296
s2  -59.730034
s3   47.492731
s4   56.203890
s5  115.537545
s6   84.554817
s7   39.580306
s8   -4.702033
s9  -54.554684
s10 -51.582594
s11 -85.953466
s12 -42.907363
> lines(sHW$fitted[,1], col= "darkgoldenrod1")
```



```
> MilkProdTSLog <- log(MilkProdTS)
> plot.ts(MilkProdTSLog)
> plot(MilkProdTSLog)
> SMATSLogS3<-SMA(SMA(MilkProdTSLog,n=3))
> SMATSLogS5<-SMA(SMA(MilkProdTSLog,n=5))
> SMATSLogS7<-SMA(SMA(MilkProdTSLog,n=7))
> SMATSLogS10<-SMA(SMA(MilkProdTSLog,n=10))
> SMATSLogS12<-SMA(SMA(MilkProdTSLog,n=12))
> lines(MilkProdTSLog,col="blue")
> lines(SMATSLogS3,col="Yellow")
> lines(SMATSLogS5,col="Red")
> lines(SMATSLogS7,col="cyan2")
> lines(SMATSLogS10,col="Black")
> lines(SMATSLogS12,col="darkmagenta")
> lines(sHWLog$fitted[,1], col= "darkgoldenrod1")
```



c. For the above, discuss how the forecasting differs in terms of MAD and MFE and why one approach or the other is better.

```
> write.csv(SMATS3, "SMATS3.csv")
> write.csv(SMATS12, "SMATS12.csv")
> write.csv(SMATS7, "SMATS7.csv")
>
> write.csv(SMATSLogS3,"SMATSLogS3.csv")
> write.csv(MilkProdTSLog,"MilkProdTSLog.csv")
```

Mean Forecast Error

$$\text{MFE} = \frac{\sum_{i=1}^{n} A_t - F_t}{n}$$

Mean Absolute Deviation

$$\text{MAD} = \frac{\sum_{i=1}^{n} |A_t - F_t|}{n}$$

Using the excel to calculate:

Month	Pounds_per_Cow	SMATS3	A-F	abs(A_F)	Log(MilkProd)	SMATSLog3	A-F	abs(A-F)
1962-01	589	NA			6.378426184	NA		
1962-02	561	NA			6.329720906	NA		
1962-03	640	NA			6.461468176	NA		
1962-04	656	NA			6.486160789	NA		
1962-05	727	NA			6.588926478	NA		
1962-06	697	NA			6.546785411	NA		
1962-07	640	NA			6.461468176	NA		
1962-08	599	NA			6.395261598	NA		
1962-09	568	NA			6.342121419	NA		
1962-10	577	NA			6.357842267	NA		
1962-11	553	NA			6.315358002	NA		
1962-12	582	623.7	-41.7	41.7	6.366470448	6.43183858	-0.06537	0.065368
1963-01	600	621.8667	-21.8667	21.86667	6.396929655	6.42881001	-0.03188	0.03188
1963-02	566	618.2333	-52.2333	52.23333	6.338594078	6.42296482	-0.08437	0.084371
1963-03	653	611.4333	41.56667	41.56667	6.481577129	6.41231633	0.069261	0.069261
1963-04	673	605.1667	67.83333	67.83333	6.51174533	6.40265113	0.109094	0.109094
1963-05	742	605.3	136.7	136.7	6.609349243	6.40283418	0.206515	0.206515
1963-06	716	611.8	104.2	104.2	6.573680167	6.41254283	0.161137	0.161137
1963-07	660	622.1667	37.83333	37.83333	6.492239835	6.42842344	0.063816	0.063816
1963-08	617	630.4667	-13.4667	13.46667	6.424869024	6.44160889	-0.01674	0.01674

MFE - SMA	8.519532909
MAD - SMA	59.24989384
MFE - LogSMA	0.011845943
MAD - LogSMA	0.077695424

1. We can observe that there is clear difference between simple moving average and exponential moving average, one is not necessarily better than the other.
2. Exponential moving average have less lag and is more sensitive.
3. Hence, exponential moving average turn before simple moving averages.
4. Simple moving average represent the true average of milk production entire time.
5. So, simple moving average can be better suited to identify support or resistance levels.

References:

1. http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:moving_averages
2. https://www.youtube.com/watch?v=DxdwsoRL9W4
3. Course Content