

PSTAT174 Project

Joseph Chang

March 11th, 2022

Data Importation

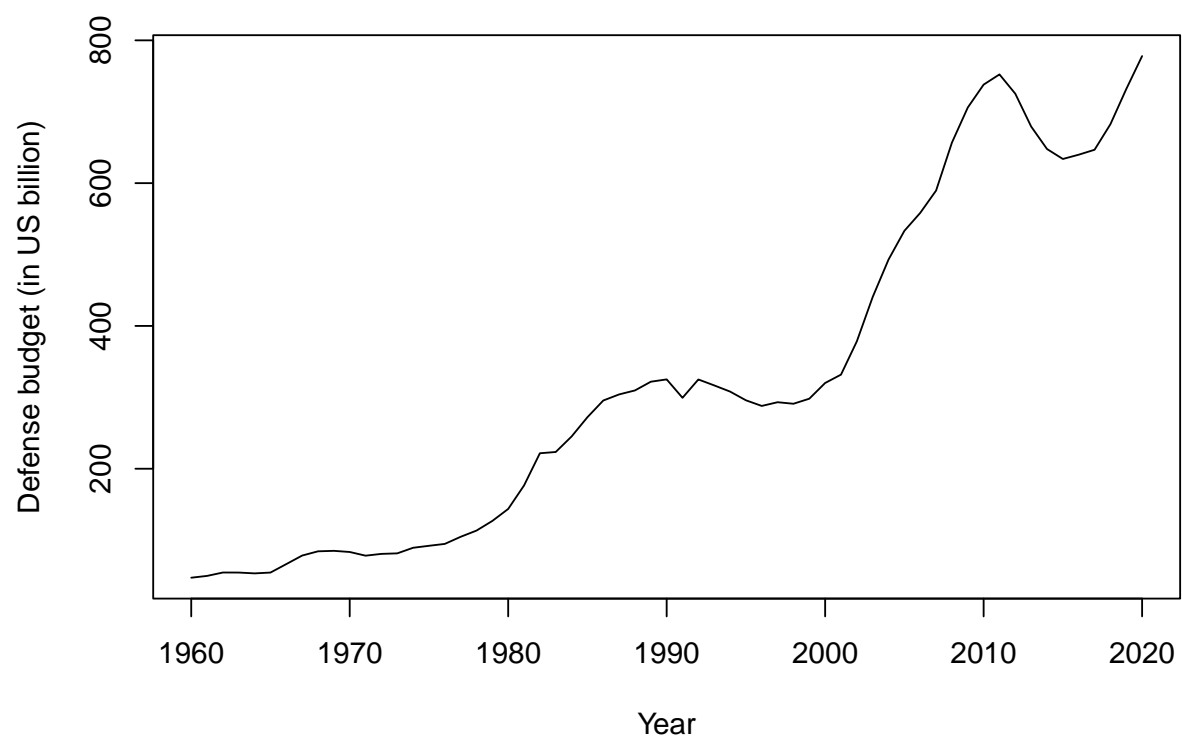
```
# load data
spending_data <- read.csv("/Users/josephchang/Desktop/MilitarySpending.csv.xls")
spending_data
```

##	Year	DefenseBudget	GDP	Population
## 1	1960	47.35	543.3	180.7
## 2	1961	49.88	563.3	183.7
## 3	1962	54.65	605.1	186.5
## 4	1963	54.56	638.6	189.2
## 5	1964	53.43	685.8	191.9
## 6	1965	54.56	743.7	194.3
## 7	1966	66.44	815.0	196.6
## 8	1967	78.40	861.7	198.7
## 9	1968	84.33	942.5	200.7
## 10	1969	84.99	1019.9	202.7
## 11	1970	83.41	1073.3	205.1
## 12	1971	78.24	1164.8	207.7
## 13	1972	80.71	1279.1	209.9
## 14	1973	81.47	1425.4	211.9
## 15	1974	89.28	1545.2	213.8
## 16	1975	92.08	1684.9	216.0
## 17	1976	94.72	1873.4	218.0
## 18	1977	104.67	2081.8	220.2
## 19	1978	113.38	2351.6	222.6
## 20	1979	126.88	2627.3	225.1
## 21	1980	143.69	2857.3	227.2
## 22	1981	176.56	3207.0	229.5
## 23	1982	221.67	3343.8	231.7
## 24	1983	223.43	3634.0	233.8
## 25	1984	245.15	4037.6	235.8
## 26	1985	272.16	4339.0	237.9
## 27	1986	295.55	4579.6	240.1
## 28	1987	304.09	4855.2	242.3
## 29	1988	309.66	5236.4	244.5
## 30	1989	321.87	5641.6	246.8
## 31	1990	325.13	5963.1	249.6
## 32	1991	299.37	6158.1	253.0
## 33	1992	325.03	6520.3	256.5

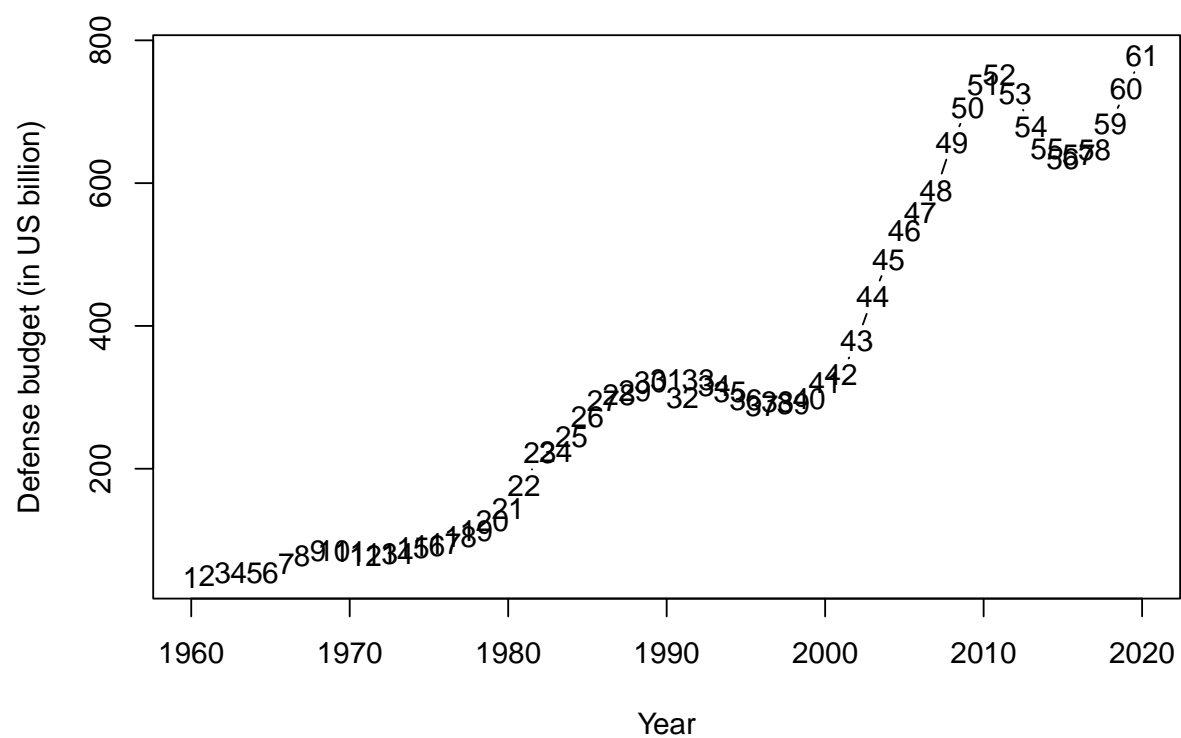
## 34 1993	316.72	6858.6	259.9
## 35 1994	308.08	7287.2	263.1
## 36 1995	295.85	7639.8	266.3
## 37 1996	287.96	8073.1	269.4
## 38 1997	293.17	8577.5	272.6
## 39 1998	291.00	9062.8	275.9
## 40 1999	298.09	9630.7	279.0
## 41 2000	320.09	10252.4	282.2
## 42 2001	331.81	10581.8	285.0
## 43 2002	378.46	10936.4	287.6
## 44 2003	440.53	11458.2	290.1
## 45 2004	493.00	12213.7	292.8
## 46 2005	533.20	13036.6	295.5
## 47 2006	558.34	13814.6	298.4
## 48 2007	589.59	14451.9	301.2
## 49 2008	656.76	14712.8	304.1
## 50 2009	705.92	14448.9	306.8
## 51 2010	738.01	14992.0	309.3
## 52 2011	752.29	15542.6	311.6
## 53 2012	725.21	16197.0	313.8
## 54 2013	679.23	16784.8	316.0
## 55 2014	647.79	17527.2	318.3
## 56 2015	633.83	18238.3	320.6
## 57 2016	639.86	18745.1	322.9
## 58 2017	646.75	19543.0	325.0
## 59 2018	682.49	20611.9	326.7
## 60 2019	731.75	21433.2	328.2
## 61 2020	778.00	20940.0	330.7

```
# plot data
```

```
plot(spending_data$Year, spending_data$DefenseBudget, ylab = "Defense budget (in US billion)", xlab = "
```



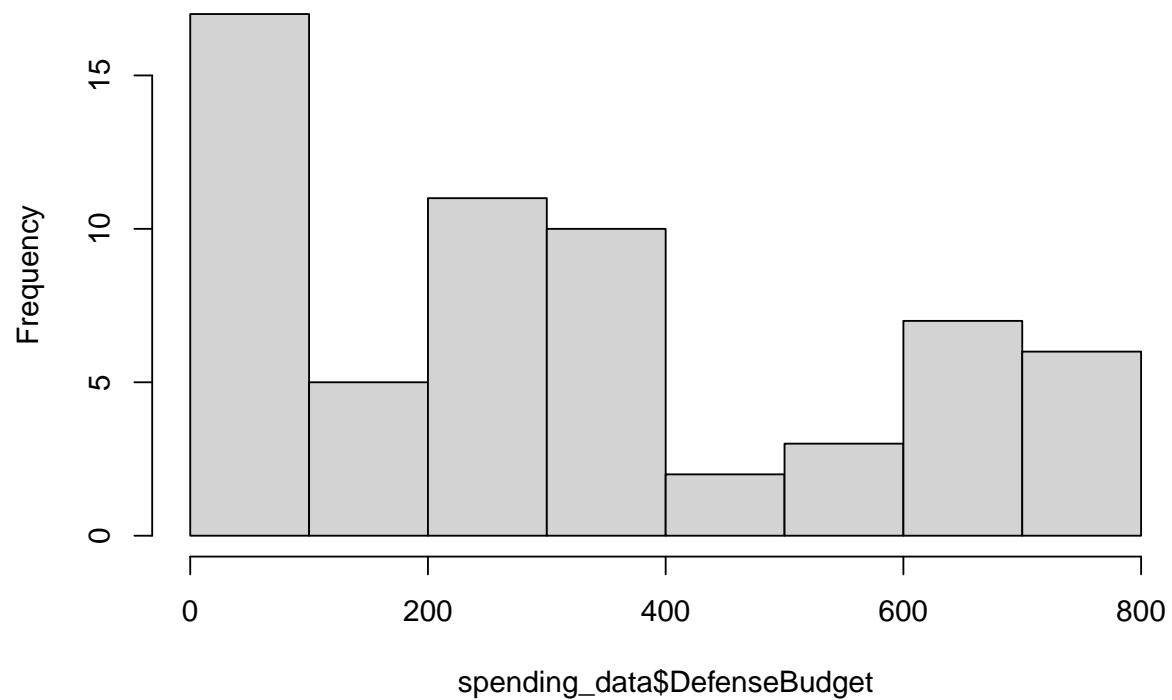
```
# plot time series  
plot.ts(spending_data$Year, spending_data$DefenseBudget, ylab = "Defense budget (in US billion)", xlab = "Year")
```



Immediate observations: There seems to be a linear trend that is positive but there is no seasonality

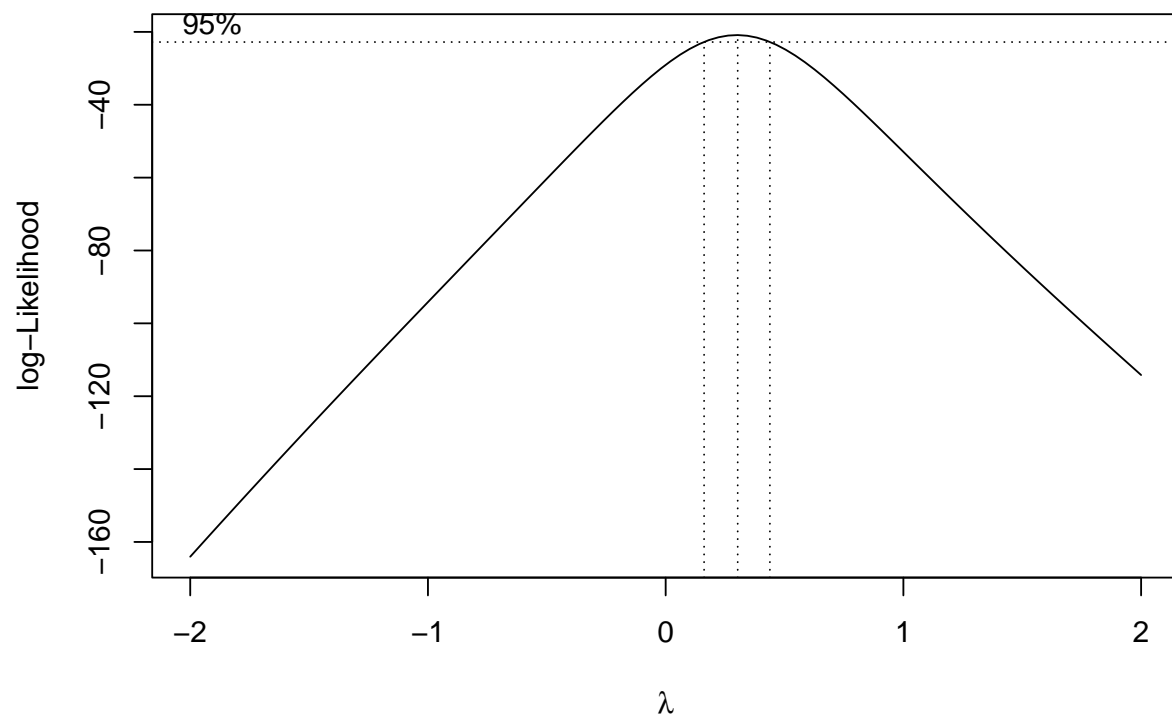
```
hist(spending_data$DefenseBudget)
```

Histogram of spending_data\$DefenseBudget



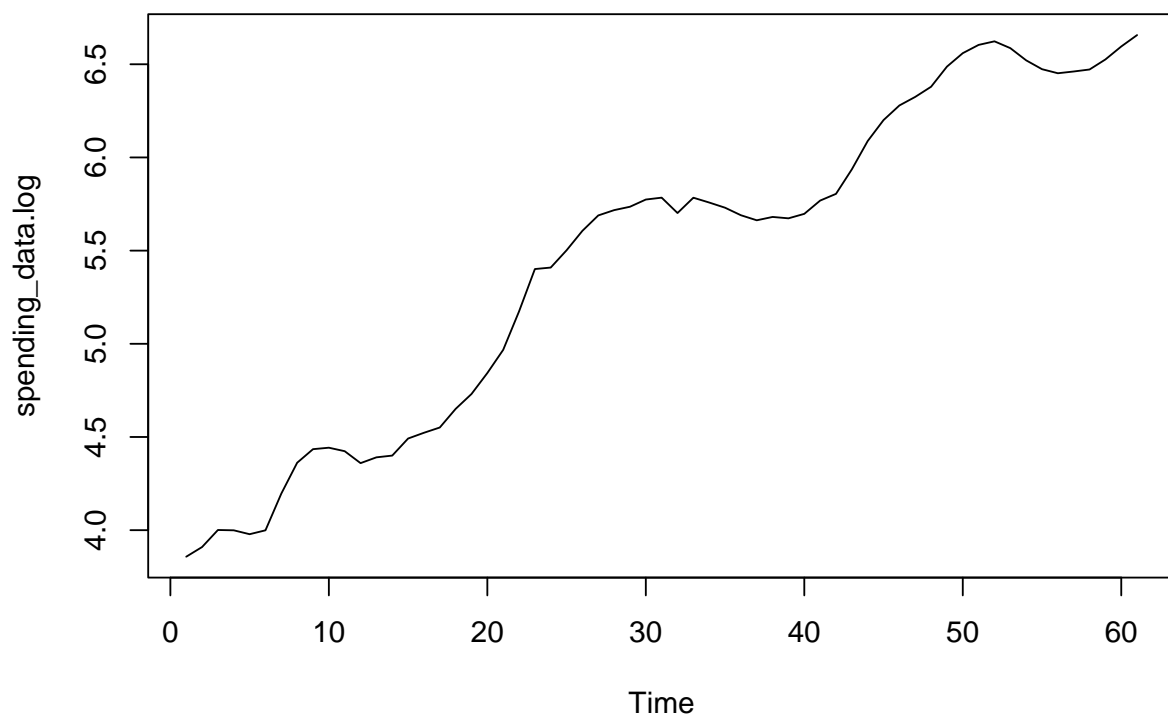
Transformation

```
# Since original data is skewed I will try Box-Cox transformation  
t <- 1:length(spending_data$DefenseBudget)  
fit <- lm(spending_data$DefenseBudget~t)  
bcTransform <- boxcox(spending_data$DefenseBudget ~ t, plotit=TRUE)
```

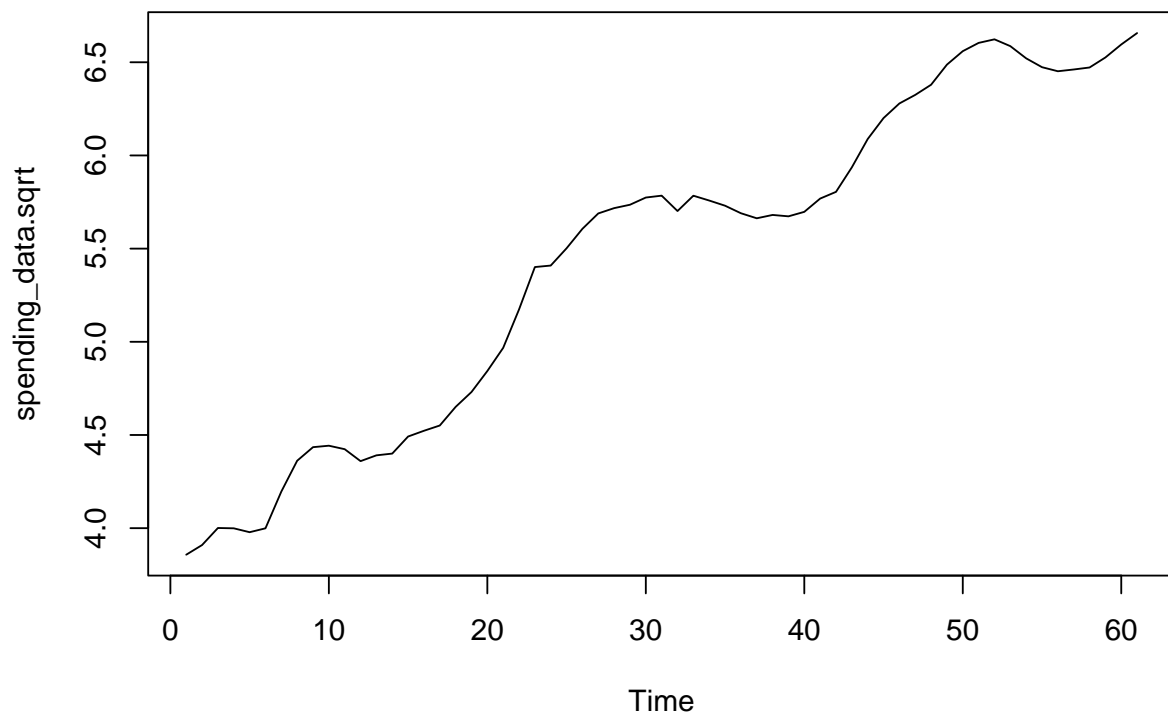


```
# best lambda
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
spent = (1/lambda)*(spending_data$DefenseBudget^lambda-1)

# comparison of best lambda vs log/sqrt transformation
spending_data.log = log(spending_data$DefenseBudget)
plot.ts(spending_data.log)
```



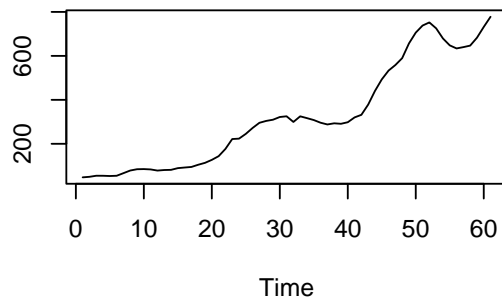
```
spending_data.sqrt = log(spending_data$DefenseBudget)
plot.ts(spending_data.sqrt)
```



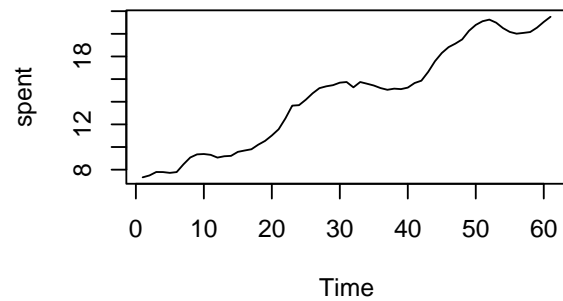
```
# compare transforms on time series plot
op=par(mfrow=c(2,2))
ts.plot(spending_data$DefenseBudget, main = "Original Time Series")
ts.plot(spent, main = "Box-cox Transform")
ts.plot(spending_data.log, main = "Log transform")
ts.plot(spending_data.sqrt, main = "Square root transform")
```


spending_data\$DefenseBudget

Original Time Series

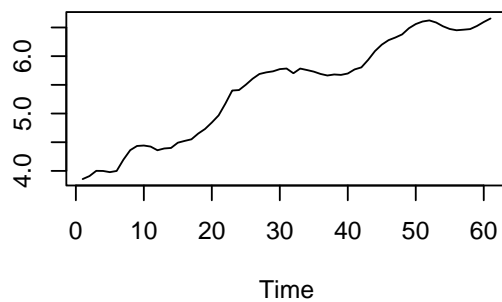


Box-cox Transform



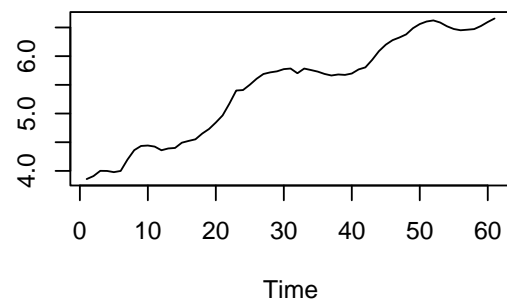
Log transform

spending_data.log



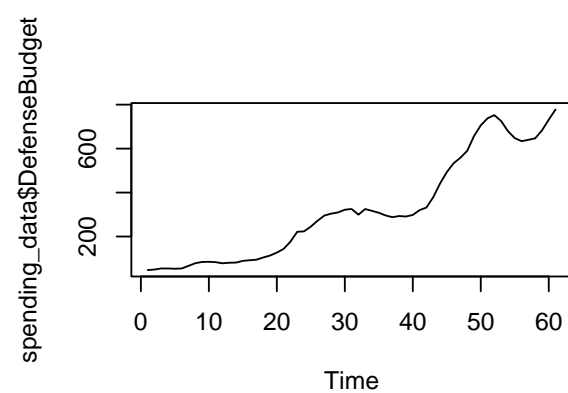
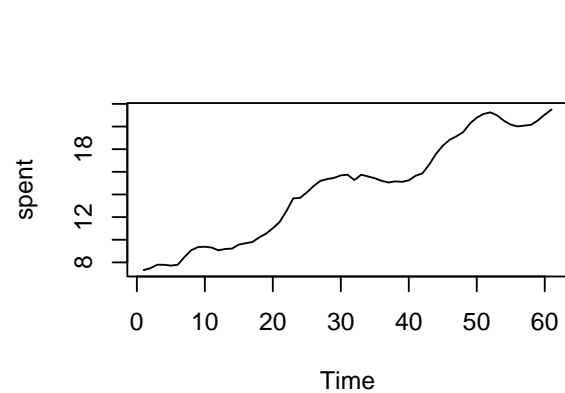
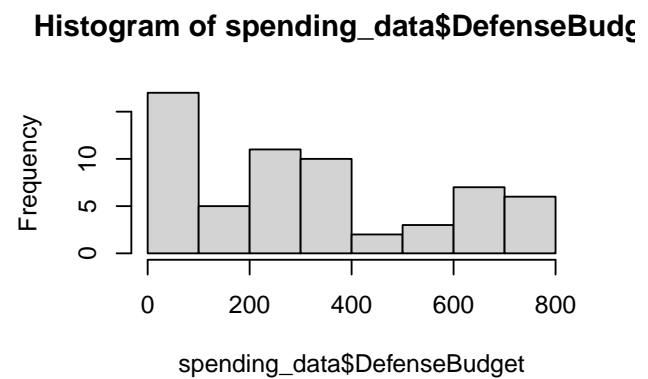
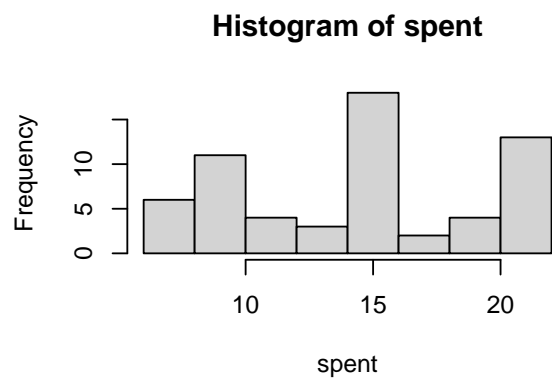
Square root transform

spending_data.sqrt



```
# histogram of spent vs original
hist(spent)
hist(spending_data$DefenseBudget)
# spent seems more normal while original is skewed

# time series plot of spent vs original
plot.ts(spent)
plot.ts(spending_data$DefenseBudget)
```



```
# both plots look around the same, but spent seems to have more spent in the median
```

```
# Based from the histogram, I will use the spent instead of the original data
```

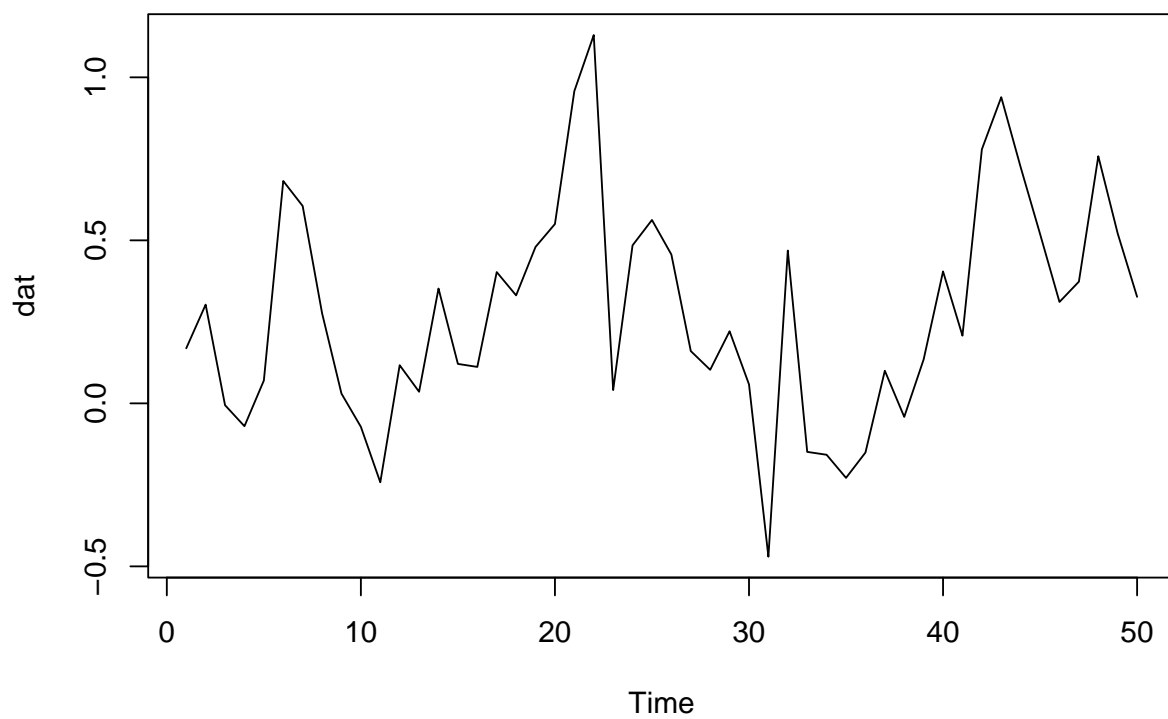
```
# training and testing dataset  
length(spent)
```

```
## [1] 61
```

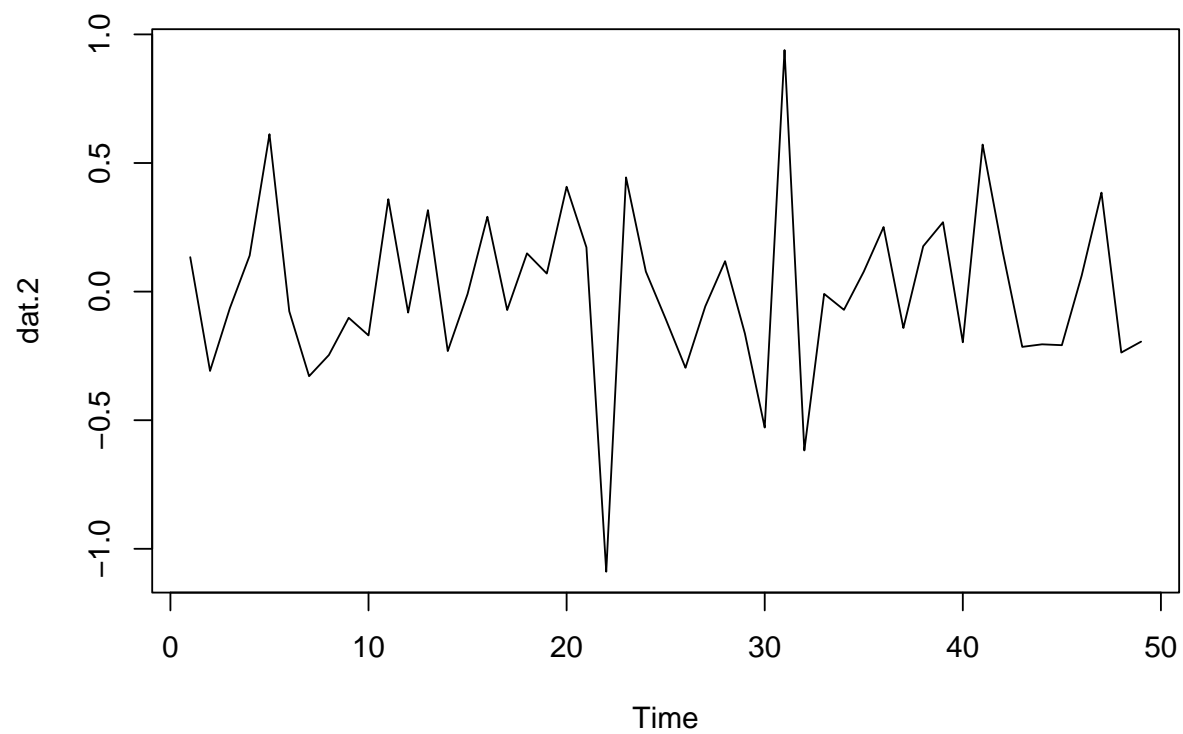
```
spent.train = spent[c(1:51)]  
spent.test  = spent[c(52:61)]
```

Differencing

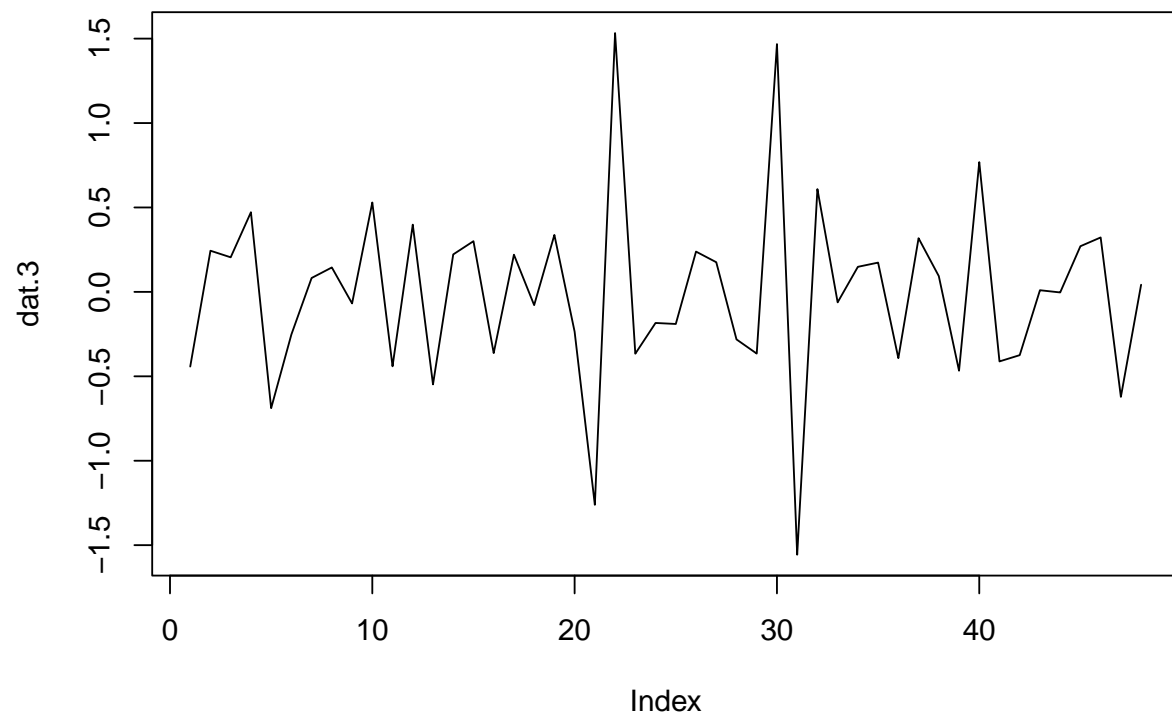
```
# difference once and plot time series  
dat <- diff(spent.train, 1)  
plot.ts(dat, type= "l")
```



```
# twice differenced and plot time series  
dat.2 <- diff(dat, 1)  
plot.ts(dat.2, type= "l")
```



```
# three times differenced and plot  
dat.3 <- diff(dat.2,1)  
plot(dat.3, type= "l")
```



```
# check variance
var(spent.train)
```

```
## [1] 15.92
```

```
var(dat)
```

```
## [1] 0.114
```

```
var(dat.2)
```

```
## [1] 0.1118
```

```
var(dat.3)
```

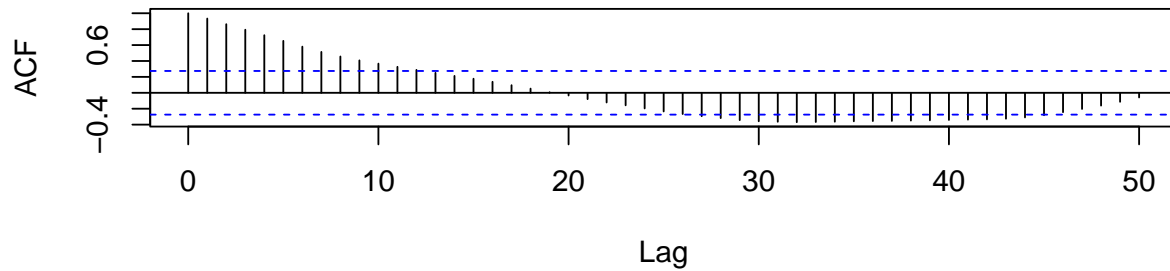
```
## [1] 0.296
```

```
# I differenced the data and found d to be 2. Based from the variance, the lowest variance is when d=2.
```

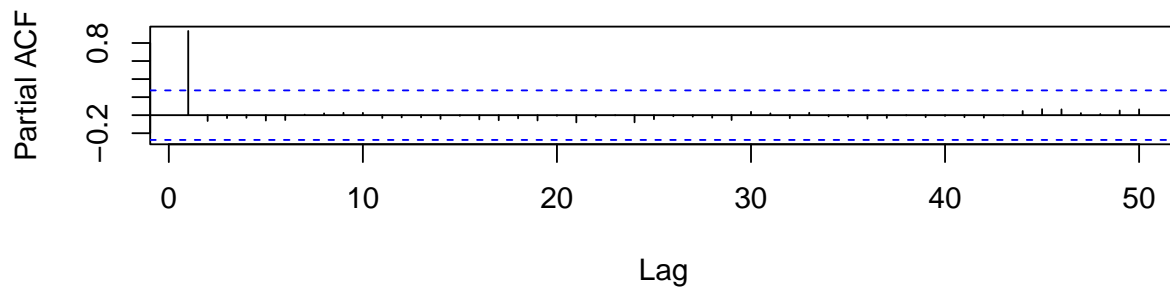
Model Identification

```
# ACF, PACF for spent
opar <- par(no.readonly = T)
par(mfrow=c(2,1))
acf(spent.train, lag.max=100)
pacf(spent.train, lag.max=100)
```

Series spent.train



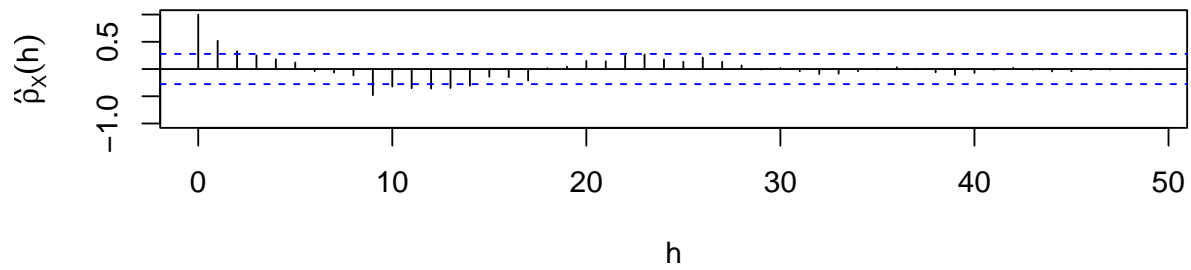
Series spent.train



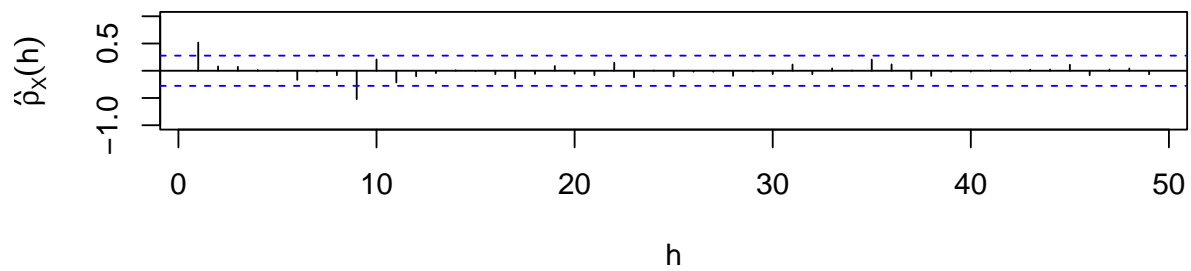
```
par(opar)

# ACF, PACF for dat (difference once)
opar <- par(no.readonly = T)
par(mfrow=c(2,1))
acf(dat, lag.max = 100, main="Sample acf", ylim=c(-1,1),xlab="h", ylab= expression(hat(rho)[X](h)))
pacf(dat, lag.max=100, main="Sample pacf", ylim=c(-1,1),xlab="h", ylab= expression(hat(rho)[X](h)))
```

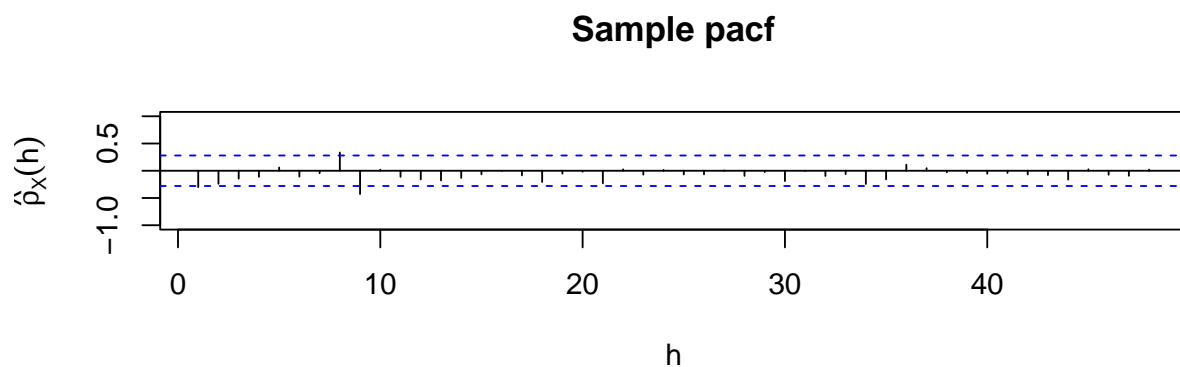
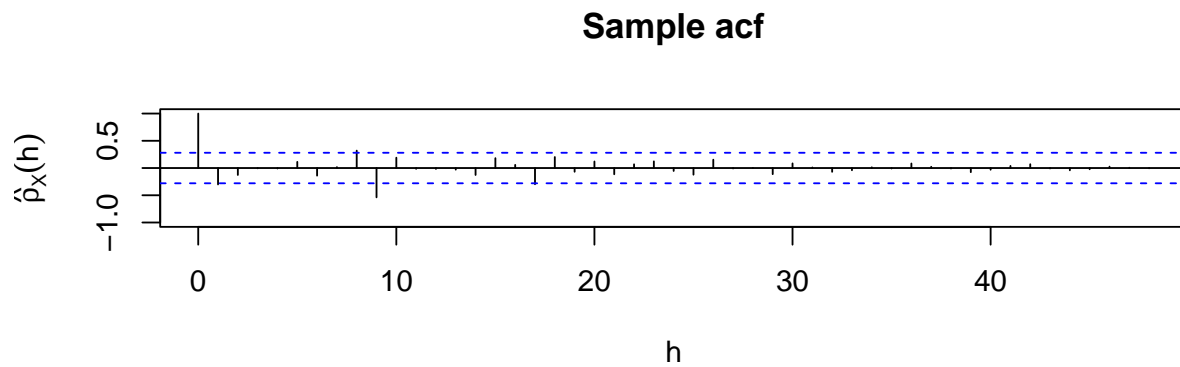
Sample acf



Sample pacf



```
# ACF, PACF for dat.2 (difference twice)
opar <- par(no.readonly = T)
par(mfrow=c(2,1))
acf(dat.2, lag.max=100, main="Sample acf", ylim=c(-1,1),xlab="h", ylab= expression(hat(rho)[X](h)))
pacf(dat.2, lag.max=100, main="Sample pacf", ylim=c(-1,1),xlab="h", ylab= expression(hat(rho)[X](h)))
```



Model estimation

```
# Candidate models:
df <- expand.grid(p=0:10, q=0:10)
df <- cbind(df, AICc=NA)

# Compute AICc:
for (i in 1:nrow(df)) {
  sarima.obj <- NULL
  try(arima.obj <- arima(spent.train, order=c(df$p[i],2, df$q[i]), method="ML"))
  if (!is.null(arima.obj)) { df$AICc[i] <- AICc(arima.obj) }
  # print(df[i, ])
}
```

```
## Warning in log(s2): NaNs produced
```

```
## Warning in log(s2): NaNs produced
```

```
## Warning in log(s2): NaNs produced
```

```
## Warning in arima(spent.train, order = c(df$p[i], 2, df$q[i]), method = "ML"):
## possible convergence problem: optim gave code = 1
```


[illegible]

```

## possible convergence problem: optim gave code = 1

## Warning in arima(spent.train, order = c(df$p[i], 2, df$q[i]), method = "ML"):
## possible convergence problem: optim gave code = 1

## Warning in arima(spent.train, order = c(df$p[i], 2, df$q[i]), method = "ML"):
## possible convergence problem: optim gave code = 1

## Warning in arima(spent.train, order = c(df$p[i], 2, df$q[i]), method = "ML"):
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in arima(spent.train, order = c(df$p[i], 2, df$q[i]), method = "ML"):
## possible convergence problem: optim gave code = 1

## Warning in arima(spent.train, order = c(df$p[i], 2, df$q[i]), method = "ML"):
## possible convergence problem: optim gave code = 1

## Error in optim(init[mask], armafn, method = optim.method, hessian = TRUE, :
##   non-finite finite-difference value [10]

## Warning in log(s2): NaNs produced

## Warning in log(s2): possible convergence problem: optim gave code = 1

df[which.min(df$AICc), ]

##      p q  AICc
## 13 1 1 26.31

df[order(df$AICc),]

##      p q  AICc
## 13  1 1 26.31
## 12  0 1 26.86
## 27  4 2 26.97
## 80  2 7 27.14
## 21  9 1 27.60
## 23  0 2 28.04
## 100 0 9 28.47
## 10  9 0 28.47
## 32  9 2 28.60
## 81  3 7 29.19
## 22 10 1 29.23
##  3  2 0 29.40
## 28  5 2 29.62
## 38  4 3 29.63
## 39  5 3 29.79
## 15  3 1 29.96
## 67  0 6 30.01

```

```

## 91  2  8 30.11
## 25  2  2 30.12
##  2  1  0 30.17
## 24  1  2 30.30
## 34  0  3 30.30
## 14  2  1 30.33
## 111 0 10 30.36
## 51  6  4 30.43
##  4  3  0 30.68
## 101 1  9 30.79
## 112 1 10 30.96
## 79  1  7 31.20
## 11 10  0 31.47
## 45  0  4 31.70
## 54  9  4 31.87
## 40  6  3 31.93
## 43  9  3 32.03
## 33 10  2 32.04
## 52  7  4 32.12
## 57  1  5 32.17
## 48  3  4 32.25
## 46  1  4 32.31
## 26  3  2 32.38
## 50  5  4 32.38
## 92  3  8 32.41
## 63  7  5 32.41
## 36  2  3 32.42
##  5  4  0 32.52
## 62  6  5 32.57
## 35  1  3 32.65
## 68  1  6 32.66
## 78  0  7 32.68
##  1  0  0 32.70
## 61  5  5 32.86
## 60  4  5 32.88
## 83  5  7 32.95
## 31  8  2 33.00
## 20  8  1 33.26
## 47  2  4 33.33
## 56  0  5 33.50
## 102 2  9 33.78
## 82  4  7 33.86
## 93  4  8 33.91
## 90  1  8 34.16
## 58  2  5 34.17
## 69  2  6 34.26
## 49  4  4 34.43
## 71  4  6 34.58
## 42  8  3 34.67
##  6  5  0 34.74
## 16  4  1 34.77
## 65  9  5 34.80
## 84  6  7 34.90
## 37  3  3 34.98

```

```

## 74 7 6 35.01
## 70 3 6 35.23
## 19 7 1 35.26
## 89 0 8 35.45
## 44 10 3 35.53
## 55 10 4 35.63
## 29 6 2 35.86
## 72 5 6 36.03
## 9 8 0 36.16
## 104 4 9 36.25
## 94 5 8 36.53
## 7 6 0 36.76
## 59 3 5 36.81
## 66 10 5 36.93
## 73 6 6 36.95
## 30 7 2 36.98
## 41 7 3 37.18
## 17 5 1 37.21
## 113 2 10 37.22
## 103 3 9 37.22
## 76 9 6 37.44
## 114 3 10 37.84
## 53 8 4 38.10
## 64 8 5 39.12
## 95 6 8 39.13
## 8 7 0 39.24
## 105 5 9 39.25
## 18 6 1 39.42
## 115 4 10 39.55
## 75 8 6 39.86
## 85 7 7 40.30
## 87 9 7 40.91
## 77 10 6 41.07
## 106 6 9 41.53
## 96 7 8 42.88
## 116 5 10 43.06
## 107 7 9 43.63
## 117 6 10 44.63
## 98 9 8 45.26
## 86 8 7 45.28
## 88 10 7 45.41
## 108 8 9 47.35
## 118 7 10 47.49
## 97 8 8 47.82
## 99 10 8 48.81
## 119 8 10 52.18
## 109 9 9 52.60
## 110 10 9 52.60
## 120 9 10 54.31
## 121 10 10 63.93

```

```

# using principle of parsimony and lowest AICC, I will consider the two best models which are (1,1) and
# Final model 1

```

```
fit1 <- arima(spent.train, order = c(1,2,1),
              method = "ML")
fit1
```

```
##
## Call:
## arima(x = spent.train, order = c(1, 2, 1), method = "ML")
##
## Coefficients:
##      ar1      ma1
##    0.539 -1.000
## s.e. 0.124 0.077
##
## sigma^2 estimated as 0.0833: log likelihood = -10.03, aic = 26.06
```

```
# Final model 2
fit2 <- arima(spent.train, order = c(0,2,1),
              method = "ML")
fit2
```

```
##
## Call:
## arima(x = spent.train, order = c(0, 2, 1), method = "ML")
##
## Coefficients:
##      ma1
##    -0.490
## s.e. 0.144
##
## sigma^2 estimated as 0.0927: log likelihood = -11.39, aic = 26.78
```

```
#source("plot.roots.R")
#plot.roots(NULL,polyroot(c(1, -0.3353, 0, -0.1612)), main="(A) roots of ma part, nonseasonal ")
```

Model Diagnostics for model 1

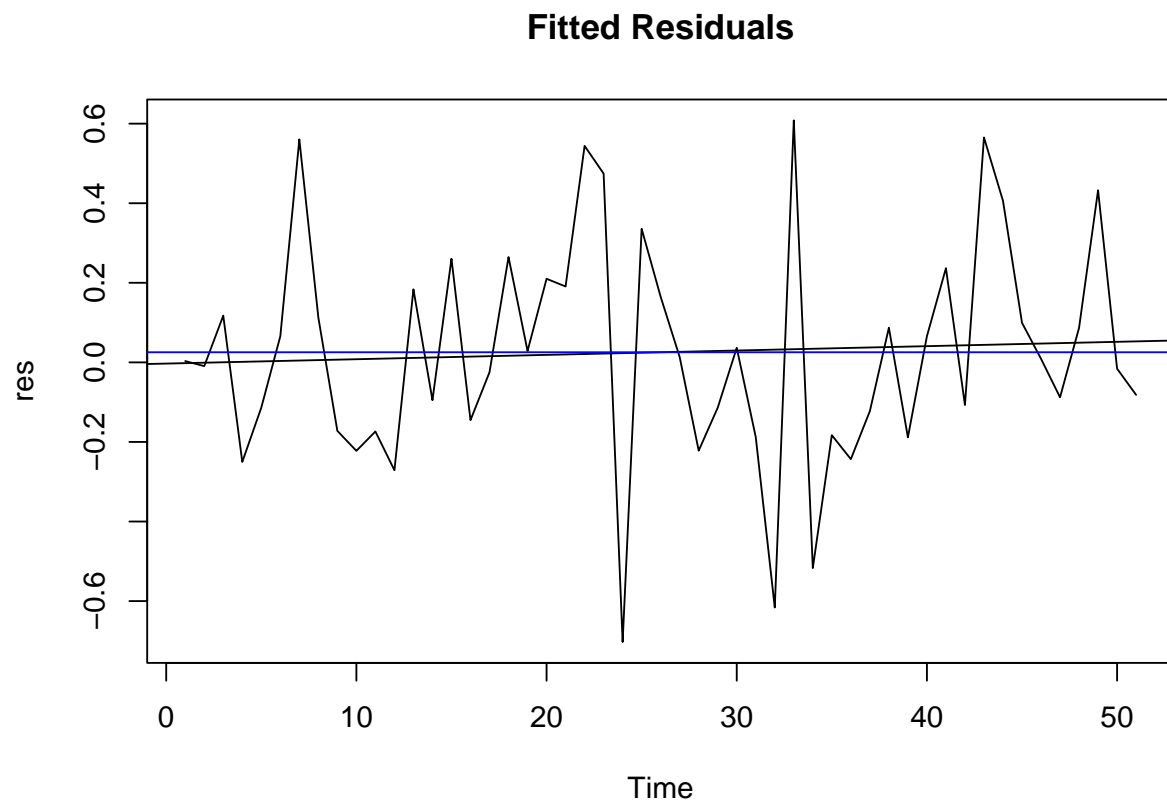
```
# residual plots
res <- residuals(fit1)
mean(res)
```

```
## [1] 0.02542
```

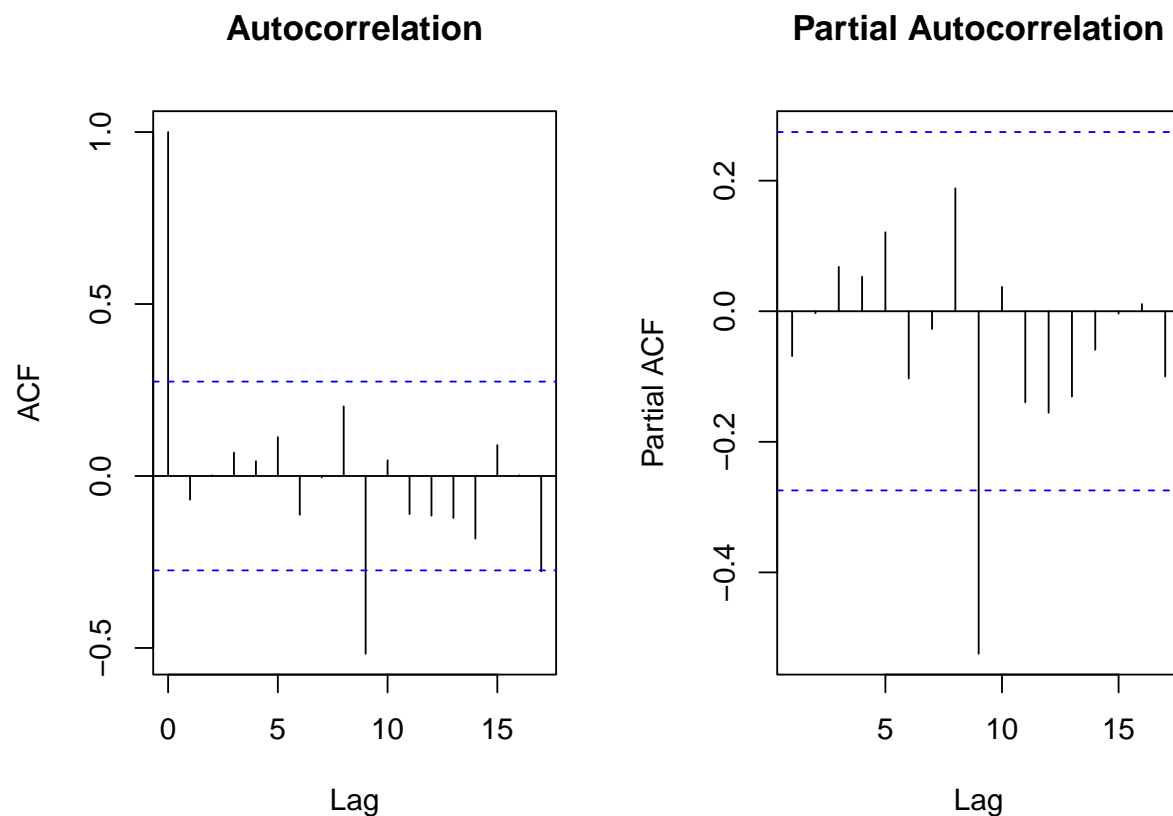
```
var(res)
```

```
## [1] 0.08103
```

```
# layout
par(mfrow=c(1,1))
ts.plot(res, main = "Fitted Residuals")
t <- 1:length(res)
fit1.res = lm(res~t)
abline(fit1.res)
abline(h=mean(res), col = "blue")
```



```
# ACF and PACF
par(mfrow=c(1,2))
acf(res, main = "Autocorrelation")
pacf(res, main = "Partial Autocorrelation")
```



```
# Testing for independence of residuals
# lag 8 is chosen since sqrt 61 is around 8
Box.test(res, lag = 8, type = c("Box-Pierce"), fitdf = 2)
```

```
##
## Box-Pierce test
##
## data:  res
## X-squared = 4, df = 6, p-value = 0.7
```

```
Box.test(res, lag = 8, type = c("Ljung-Box"), fitdf = 2)
```

```
##
## Box-Ljung test
##
## data:  res
## X-squared = 4.7, df = 6, p-value = 0.6
```

```
Box.test(res^2, lag = 8, type = c("Ljung-Box"), fitdf = 0)
```

```
##
## Box-Ljung test
##
```

```
## data:  res^2
## X-squared = 20, df = 8, p-value = 0.01
```

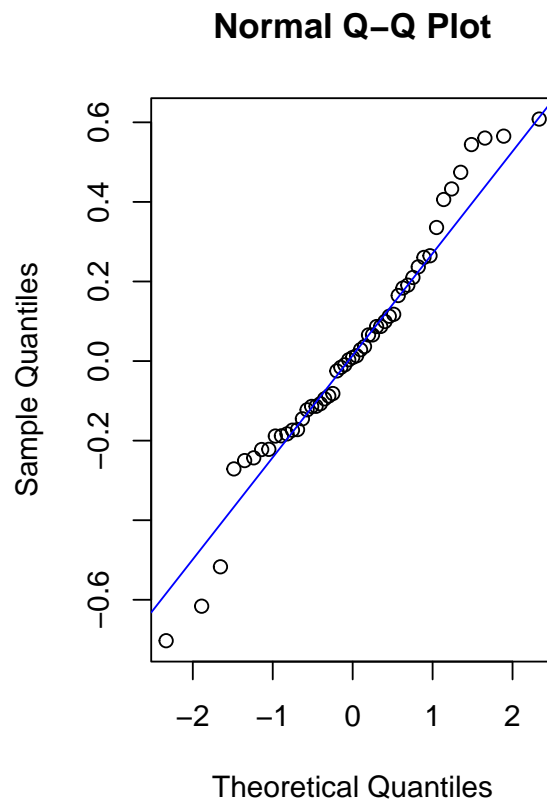
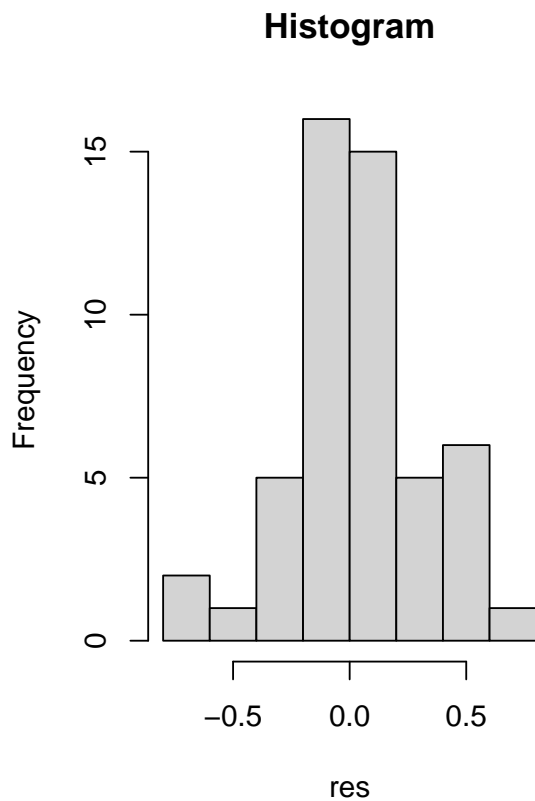
```
# test for normality of residuals
shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.97, p-value = 0.2
```

```
# yule-walker test
ar(res, aic=TRUE, order.max = NULL, method=c("yule-walker"))
```

```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
## Coefficients:
##      1      2      3      4      5      6      7      8      9
## 0.035  0.015  0.004  0.102  0.125 -0.074 -0.003  0.155 -0.525
##
## Order selected 9  sigma^2 estimated as  0.0665
```

```
# Histogram and qq plot
par(mfrow=c(1,2))
hist(res, main= "Histogram")
qqnorm(res)
qqline(res, col = "blue")
```

Model 1 passes all tests but Mc-Leod Li test and residuals are normal

Model Diagnostics for model 2

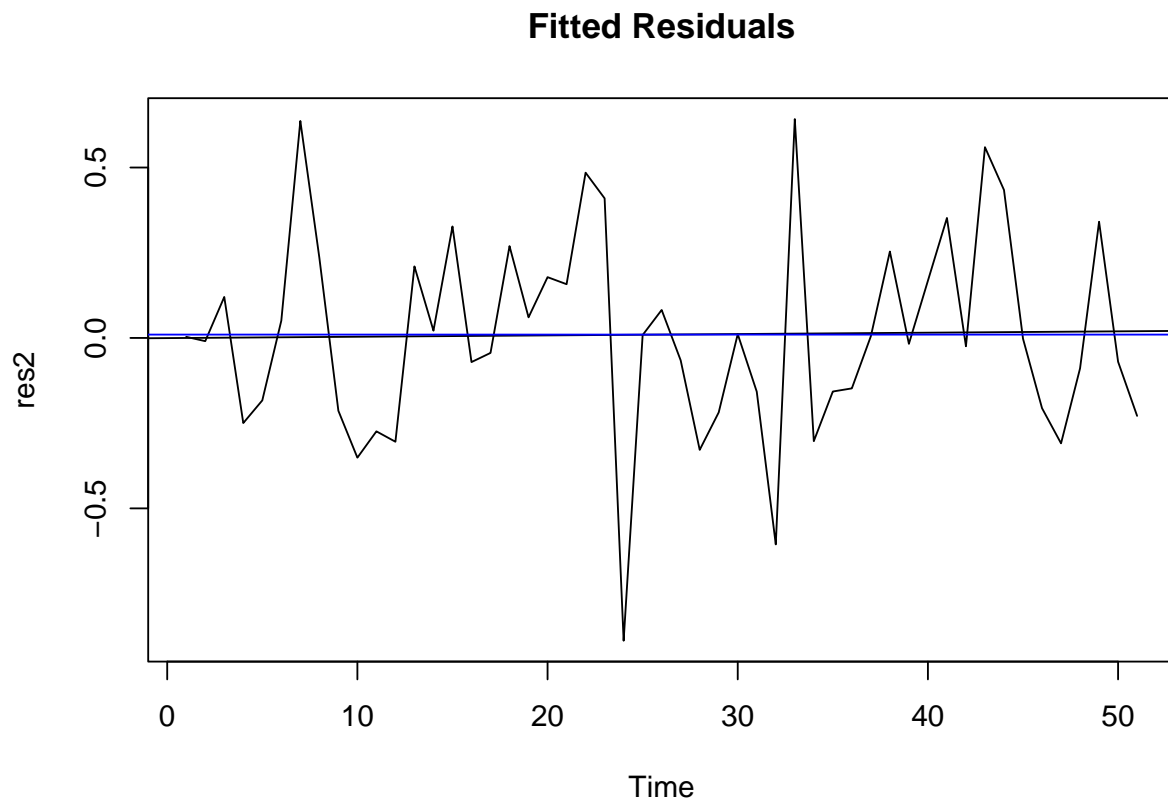
```
# residual plots
res2 <- residuals(fit2)
mean(res2)
```

```
## [1] 0.009866
```

```
var(res2)
```

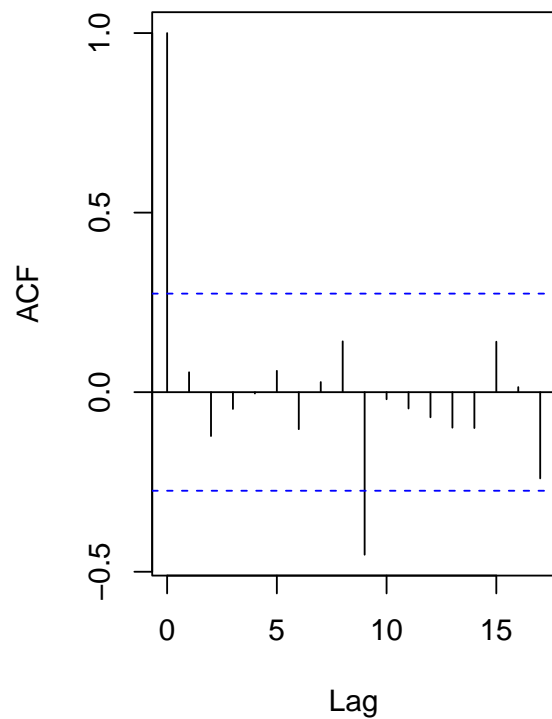
```
## [1] 0.09073
```

```
# layout
par(mfrow=c(1,1))
ts.plot(res2, main = "Fitted Residuals")
t <- 1:length(res2)
fit2.res2 = lm(res2~t)
abline(fit2.res2)
abline(h=mean(res2), col = "blue")
```

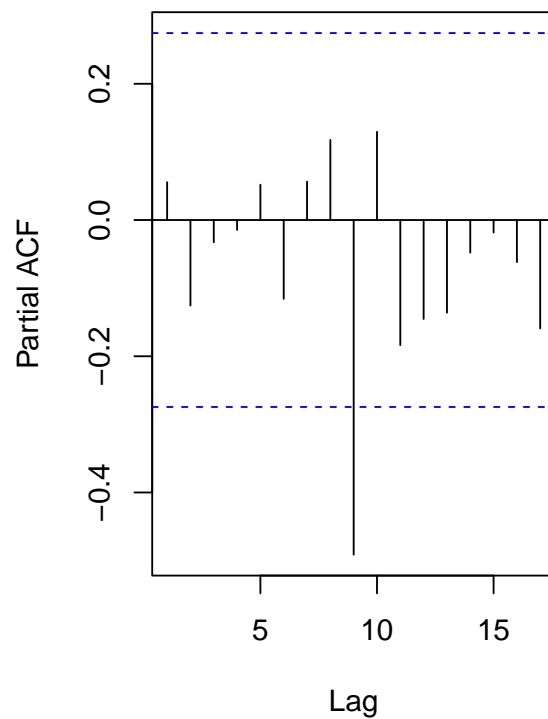


```
# ACF and PACF  
par(mfrow=c(1,2))  
acf(res2, main = "Autocorrelation")  
pacf(res2, main = "Partial Autocorrelation")
```

Autocorrelation



Partial Autocorrelation



```
# Testing for independence of residuals
```

```
Box.test(res2, lag = 8, type = c("Box-Pierce"), fitdf = 1)
```

```
##
## Box-Pierce test
##
## data:  res2
## X-squared = 2.8, df = 7, p-value = 0.9
```

```
Box.test(res2, lag = 8, type = c("Ljung-Box"), fitdf = 1)
```

```
##
## Box-Ljung test
##
## data:  res2
## X-squared = 3.3, df = 7, p-value = 0.9
```

```
Box.test(res2^2, lag = 8, type = c("Ljung-Box"), fitdf = 0)
```

```
##
## Box-Ljung test
##
## data:  res2^2
## X-squared = 9.4, df = 8, p-value = 0.3
```

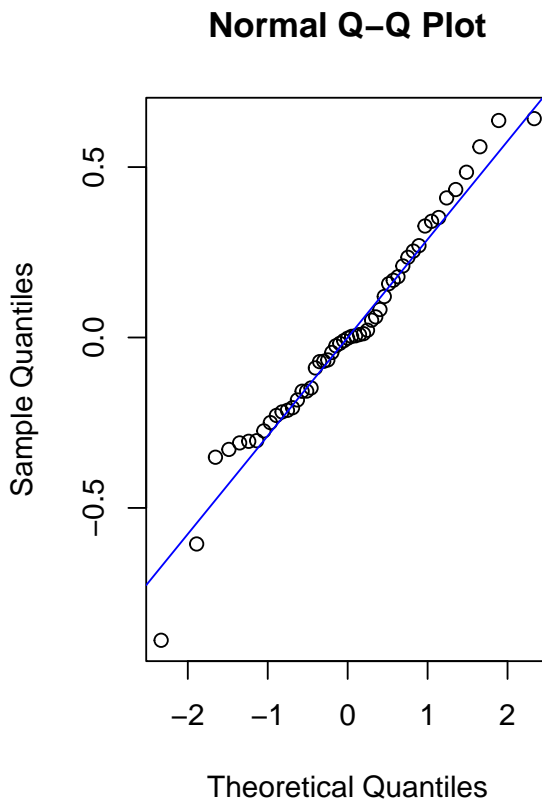
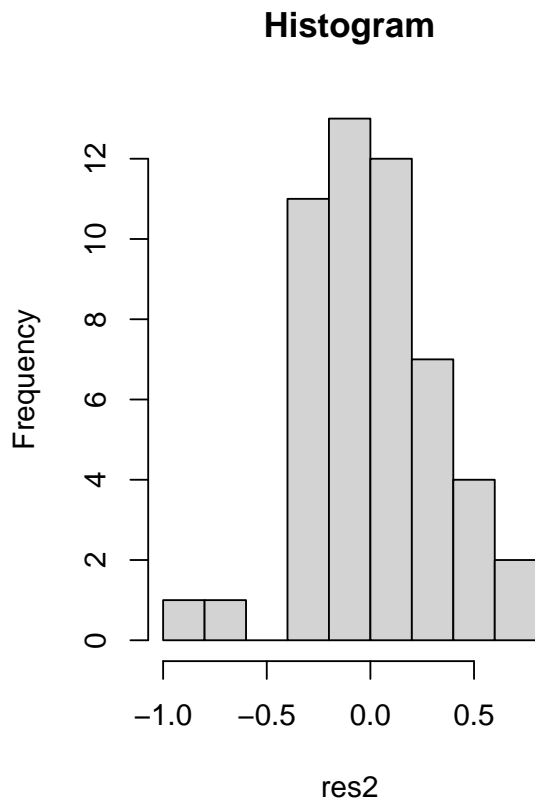
```
# test for normality of residuals
shapiro.test(res2)
```

```
##
## Shapiro-Wilk normality test
##
## data:  res2
## W = 0.98, p-value = 0.4
```

```
# yule-walker test
ar(res2, aic=TRUE, order.max = NULL, method=c("yule-walker"))
```

```
##
## Call:
## ar(x = res2, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  0.0907
```

```
# Histogram and qq plot
par(mfrow=c(1,2))
hist(res2, main= "Histogram")
qqnorm(res2)
qqline(res2, col = "blue")
```



```
# residuals pass Box.test and residuals are normal
```

```
# I will use fit2 as the final model  
fit2
```

```
##  
## Call:  
## arima(x = spent.train, order = c(0, 2, 1), method = "ML")  
##  
## Coefficients:  
##          ma1  
##        -0.490  
## s.e.    0.144  
##  
## sigma^2 estimated as 0.0927:  log likelihood = -11.39,  aic = 26.78
```

Final model should be $(1-B^2)X_t = (1-0.49B)Z_t$, $Z_t \sim WN(0, 0.0927)$

Data Forecasting

```
# Predict 10 future observations and plot  
par(mfrow=c(1,1))  
m <- length(spent.train)  
mypred <- predict(fit2, n.ahead=10)  
ts.plot(c(lambda*spent.train +1)^(1/lambda), xlim=c(1, length(lambda*spent.train)+10), ylim=c(100,1200))  
  
points((m+1):(m+10), col="red", (lambda*mypred$pred +1)^(1/lambda))  
points((m+1):(m+10), col="blue", (lambda*spent.test +1)^(1/lambda))  
lines((m+1):(m+10),  
      (lambda*mypred$pred + 1.96*mypred$se+1)^(1/lambda), lty=2)  
lines((m+1):(m+10),  
      (lambda*mypred$pred - 1.96*mypred$se+1)^(1/lambda), lty=2)
```

