ARMA using R

Time series plots

```
#Band plot
library(gplots);
x=1:1000; y=rnorm(1000, mean=1, sd=1+x/500);
x11(); bandplot(x, y, main='Band Plot');
legend('bottomleft' , c('m+/- 2d', 'm+/- d', 'mean'), col=c('magenta', 'blue', 'red'), lwd=c(2,2,2), cex=0.6);


#Example
ind=labels(AirPassengers);
x11(); bandplot(ind, AirPassengers, main='Band Plot');

x11(); seasonplot(AirPassengers, col=rainbow(12), year.labels=T);
x11(); ggseasonplot(AirPassengers, year.labels=T, continuous=T);
```

Prediction (random walk)

```
# Box-Cox transformation
library(forecast);
x11(); plot(AirPassengers, main='Air Passengers');  # original scale
(lambda=BoxCox.lambda(AirPassengers));
new=BoxCox(AirPassengers, lambda);
x11(); plot(new, main='BoxCox Air Passengers');               # scale-changed
x11(); plot(diff(new), main='Diff-BoxCox Air Passengers');

#prediction & evaluation (random walk forecast)
library(forecast);
(rf=rwf(AirPassengers));     # point forecast
x11(); plot(rf);   #prediction interval (80%, 95% upper/lower ends)
accuracy(rf);

# white noise
set.seed(1234); w=rnorm(300); wt=ts(w);
x11(); plot(wt, xlab='time', main='white noise'); abline(h=0);
x11(); acf(wt);
x11(); pacf(wt);
```

Correcting missing value & outlier

```
# estimation of missing values(multiple imputation by chained equation)
# multiple imputation: Monte Carlo simulation based data merge(random)
x=c(22, 34, NA, 36, 28, 35, 46, 42, 39, 25, 36, 25, 38, NA, 37); t=seq(1:15);
library(mice);
z=mice(matrix(c(x, t), nc=2));
t(complete(z));

# fill the missing values
library(forecast); data(gold);
table(is.na(gold));                    # check the number of missing values
for(i in 1: length(gold))              # check the location of missing values
   ifelse(is.na(gold[i]), print(i), next);

tsoutliers(gold);                      # check the location of outliers
gold[tsoutliers(gold)$index];          # check the outliers
newgold=tsclean(gold);                 # correct missing values and outliers
x11(); tsdisplay(gold);                # graph before correction
x11(); tsdisplay(newgold);             # graph after correction
```

R code for SACF and SPACF

```
library(forecast); library(tseries);

# AR(1) model
x=w=rnorm(500);
for(t in 2:300) x[t]=0.7*x[t-1]+w[t]; xt=ts(x);
x11(); par(mfrow=c(1,2)); acf(xt); pacf(xt);

# ARMA(1, 1) model
x=arima.sim(n=500, list(ar=0.7, ma=0.2)); xt=ts(x);
x11(); par(mfrow=c(1,2)); acf(xt); pacf(xt);

# ARIMA(1, 1, 1) model
x=arima.sim(n=500, list(order=c(1, 1, 1), ar=0.7, ma=0.2)); xt=ts(x);
x11(); par(mfrow=c(1,2)); acf(xt); pacf(xt);

# ARIMA(2, 1, 1) model
x=arima.sim(n=500, list(order=c(2, 1, 1), ar=c(0.7, 0.2), ma=0.2)); xt=ts(x);
x11(); par(mfrow=c(1,2)); acf(xt); pacf(xt);

# Random Walk
ddd=arima.sim(list(order=c(0, 1, 0)), n=500); x11(); plot(ddd);
x11(); par(mfrow=c(1,2)); acf(ddd); pacf(ddd);
```

Unit Root Tests

```
library(tseries);
### stationary test: unit root test #####
kpss.test(AirPassengers);     # H0: stationary
# check stationarity after deleting the trend
x11();plot(AirPassengers); kpss.test(AirPassengers, 'Trend');
pp.test(AirPassengers);    # H0: non-stationary(unit-root)
adf.test(AirPassengers);    # H0: non-stationary(unit-root)
######
x=rnorm(1000); y=diffinv(x); # x has no unit-root, but y contains a unit-root.
adf.test(x, k=3); pp.test(x); kpss.test(x);
############ more advanced approach
y01=read.delim("mraw.txt", header=T);
m1w1=lm(wti~-1+wti1+snp, data=y01);
re1=residuals(m1w1);
library(urca); ut1=ur.df(re1, type= "none", selectlags="AIC");  # lag selection based on AIC criteria
summary(ut1);  # if z.lag.1 is not significant(coefficient of lag1 y=0) , there is unit root
?ur.df
############
library(tseries); adf.test(re1);
adf.test(re1, k=2);
########
data(Raotbl3)
attach(Raotbl3)
lc.df <- ur.df(y=lc, lags=3, type='trend')    # specify the lag
summary(lc.df)
```

Autocorrelation Tests

- Runs test, Portmanteau test: Box test (Box-Pierce, Ljung-Box)

```
### Runs test
library(tseries);
x=rnorm(50); x1=factor(ifelse(x>=median(x), 1, 0));
runs.test(x1, a='less');


### Portmanteau test: Box test (Box-Pierce, Ljung-Box)
Box.test(x, t=c('B'));  # Box-Pierce test
Box.test(x, t=c('L'))   # Ljung-Box test


### arima.sim & arima ##########
y=arima.sim(list(order=c(1,0,0), ar=0.4), n=300);  #AR(1) simulation
x11(); ts.plot(y);
a=arima(y,order=c(1,0,0));  # AR(1) estimation
?arima   # see method
forecast(a, h=20);
predict(a, n.ahead=20);
Box.test(a$residuals, lag=2);
Box.test(a$residuals, lag=2, type="Ljung-Box")
```

Co-integration Test & Granger Causality

```
### Phillips-Ouliars Co-integration test
x=diffinv(rnorm(1000)); y=2-3*x+rnorm(x, sd=5);
z=ts(cbind(x,y));            # x and y are co-integrated
x11(); plot(z);
po.test(z);  # null: no co-integration

### Johansen Co-integration test: useful for vector AR
data(denmark) ;
sjd <- denmark[, c("LRM", "LRY", "IBO", "IDE")] ;
head(sjd);

sjd.vecm <- ca.jo(sjd, ecdet = "const", type="eigen", K=2, spec="longrun", season=4) ;
summary(sjd.vecm);
?ca.jo

########################## Granger Causality test
library(lmtest); data(ChickEgg);
grangertest(chicken~egg, order=3, data=ChickEgg);  # egg granger-caused chicken
grangertest(egg~chicken, order=3, data=ChickEgg); # chicken did not granger-cause egg

# alternative way to give same result
grangertest(ChickEgg, order=3);
grangertest(ChickEgg[,1], ChickEgg[,2], order=3);
```

AR model estimation

```
library(forecast); library(tseries);
############## Example 1 (no trend) ##############
dd1=c(1342, 1442, 1252, 1343, 1425, 1362, 1456, 1272, 1243, 1359, 1412, 1253, 1201, 1478, 1322,
1406, 1254, 1289, 1497, 1208);
d1=ts(dd1, start=c(2016, 1), frequency=4);

# stationary test (unit-root test)
ddt=d1; kpss.test(ddt); kpss.test(ddt, 'Trend');
x11(); tsdisplay(ddt, main='time series display');

# ar w/ Yule-walker estimation
(ar1=ar(ddt, m=c('yule-walker'))); # AR(2) fitting
(f1=forecast(ar1));
accuracy(f1);

x11(); plot(f1, xlab='time', ylab='series'); abline(h=mean(ddt));
grid(); title('₩n ₩n estimation: yule-walker');
Box.test(ar1$resid, type='Ljung-Box');
temp1=window(ar1$resid, start=c(2016, 3));
jarque.bera.test(temp1);
```

AR model

```
# ar w/ OLS estimation
(ar3=ar(ddt, m=c('ols'))); # AR(8) fitting
(f3=forecast(ar3));
accuracy(f3);

x11(); plot(f3, xlab='time', ylab='series'); abline(h=mean(ddt)); grid(); title('₩n  ₩n  estimation: ols');
Box.test(ar3$resid, type='Ljung-Box');
temp3=window(ar3$resid, start=c(2018, 1));
jarque.bera.test(temp3);

# ar w/ MLE estimation
(ar4=ar(ddt, m=c('mle'))); # AR(8) fitting
(f4=forecast(ar4));
accuracy(f4);

x11(); plot(f4, xlab='time', ylab='series'); abline(h=mean(ddt)); grid(); title('₩n  ₩n  estimation: mle');
Box.test(ar4$resid, type='Ljung-Box');
temp4=window(ar4$resid, start=c(2018, 1));
jarque.bera.test(temp4);

# fpeaut: AR model optimal lag selection
library(timsac);
fpeaut(ddt)$order;
fpeaut(ddt)$best.ar;  # AR(2)
```

ARMA model

```
library(tseries);  # arma function
############ Example 1 ############
dam=c(1142, 1242, 1252, 1343, 1225, 1562, 1365, 1572, 1343, 1459, 1412, 1453, 1401, 1478, 1322,
1606, 1554, 1589, 1597, 1408);
damt=ts(dam, start=c(2016,1), frequency=4); kpss.test(damt); kpss.test(damt, 'Trend');
x11(); tsdisplay(damt, main='time series display');

#ARMA(2, 0) model
arm2=arma(damt, order=c(2,0)); summary(arm2); fitted(arm2);
x11(); plot(damt, type='b', main='ARMA (2, 0)'); lines(fitted(arm2), col='red', lty=6, lwd=2); grid();

#ARMA(0, 1) model
arm3=arma(damt, order=c(0,1)); summary(arm3); fitted(arm3);
x11(); plot(damt, type='b', main='ARMA (0, 1)'); lines(fitted(arm3), col='red', lty=6, lwd=2); grid();

#ARMA(1, 1) model
arm11=arma(damt, order=c(1,1)); summary(arm11); fitted(arm11);
x11(); plot(damt, type='b', main='ARMA (0, 2)'); lines(fitted(arm11), col='red', lty=6, lwd=2); grid();

# autoarmafit: ARMA model lag selection
library(timsac);
(aa=autoarmafit(damt));  # Best ARMA model search
ar=aa$model[[1]]$arcoef; ma=aa$model[[1]]$macoef; va=aa$model[[1]]$v;
x11(); prdctr(damt, r=5, s=21, h=4, arcoef=ar, macoef=ma, v=va); title('₩n  ₩n  Auto ARMA');
```

ARIMA model

```
########### Example 1 ###########
ddd=arima.sim(list(order=c(0, 1, 0)), n=500);
auto.arima(ddd, allowdrift=F);   # optimal model: ARIMA(0, 1, 0)
am=arima(ddd, order=c(0, 1, 0)); x11(); tsdiag(am);

# auto.arima: ARIMA model lag selection
########### Example 2 (no trend) ###########
dd1=c(1342, 1442, 1252, 1343, 1425, 1362, 1456, 1272, 1243, 1359, 1412, 1253, 1201, 1478, 1322,
1406, 1254, 1289, 1497, 1208);
d1=ts(dd1, start=c(2016, 1), frequency=4);
am1=auto.arima(d1, allowdrift=F);   # optimal model: ARIMA(0, 0, 1)
x11(); tsdiag(am1);
fitted(am1); forecast(am1);
x11(); plot(forecast(am1)); lines(fitted(am1), col='red', lty=2, lwd=2);

########### Example 3 (w/ drift) ###########
dd2=c(1142, 1242, 1452, 1543, 1225, 1362, 1556, 1672, 1343, 1459, 1662, 1753, 1421, 1558, 1772,
1846, 1554, 1649, 1877, 1948);
d2=ts(dd2, start=c(2016, 1), frequency=4);
am2=auto.arima(d2);   # optimal model: ARIMA(0, 0, 0)(1, 1, 0)[4] with drift
x11(); tsdiag(am2);
fitted(am2); forecast(am2);
x11(); plot(forecast(am2)); lines(fitted(am2), col='red', lty=2, lwd=2);
```

AR(p)I(d)MA(q) model

```
library(forecast); library(tseries);

### AR(ar_lag1, no_intercept)
y=arima.sim(list(order=c(1,0,0), ar=0.7), n=300);  #AR(1) simulation
aa=arima(y,order=c(1,0,0))
# AR estimation: fixed=c(ar1, intercept)
a=arima(y,order=c(1,0,0), fixed=c(NA, 0), transform.pars=F); #NA: value to estimate, 0: deleted (forced)

### AR(ar_lag2, ar_lag3)
y0=arima.sim(list(order=c(3,0,0), ar=c(0, 0.7, 0.2)), n=300);
# AR estimation: fixed=c(ar1, ar2, ar3, intercept)
a0=arima(y0,order=c(3,0,0), fixed=c(0, NA, NA, NA), transform.pars=F);

### ARMA(ar_lag3, ma_lag1, ma_lag2)
y1=arima.sim(list(order=c(3,0,2), ar=c(0, 0, 0.2), ma=c(0.4, 0.2)), n=300);
# AR estimation: fixed=c(ar1, ar2, ar3, ma1, ma2, intercept)
a1=arima(y1, order=c(3,0,2), fixed=c(0, 0, NA, NA, NA, NA), transform.pars=F);

### ARIMA(ar_lag3, ma_lag1, ma_lag2): If differenced, there is no intercept
y2=arima.sim(list(order=c(3,1,2), ar=c(0, 0, 0.2), ma=c(0.4, 0.2)), n=300);
# AR estimation: fixed=c(ar1, ar2, ar3, ma1, ma2)
a2=arima(y2, order=c(3,1,2), fixed=c(0, 0, NA, NA, NA), transform.pars=F);
```

SARIMA model

```
library(forecast);
################ Example 1 ##############
x11(); plot(decompose(USAccDeaths));
x11(); tsdisplay(USAccDeaths, main='US Accidental Deaths / Month');
findfrequency(USAccDeaths);
am0=auto.arima(USAccDeaths); summary(am0);
x11(); tsdiag(am0);
fitted(am0); forecast(am0, h=30);

################ Example 2 (w/ trend) ##############
dd2=c(1142, 1242, 1452, 1543, 1225, 1362, 1556, 1672, 1343, 1459, 1662, 1753, 1421, 1558, 1772, 1846,
1554, 1649, 1877, 1948);
d2=ts(dd2, start=c(2016, 1), frequency=4);
kpss.test(d2); kpss.test(d2, 'Trend');
findfrequency(d2);

am=auto.arima(d2); summary(am);    # optimal model: ARIMA(0, 0, 0)(1, 1, 0)[4] with drift
x11(); tsdiag(am);
fitted(am); forecast(am);
x11(); plot(forecast(am)); lines(fitted(am), col='red', lty=2, lwd=2);
```

Multiplicative Seasonal ARIMA model

- Co2 data: Monthly Carbon Dioxide Levels at Alert, NWT, Canada

```
data(co2);
x11();   # 1959/1~1997/12
win.graph(width=4.875, height=3, pointsize=8);
plot(co2, ylab='CO2');

# plot with month symbol for subset data
x11();
plot(window(co2, start=c(1995,1)), ylab='CO2');
Month=c('Ja','F','M','A','My','Jn','J','Au','S','O','N','D');
points(window(co2, start=c(1995,1)), pch=Month);

#sample ACF of co2
x11(); pacf(as.vector(co2), lag.max=36);

# 1st order difference of co2
x11(); plot(diff(co2), ylab='1st order difference of co2', xlab='time');
x11(); pacf(as.vector(diff(co2)), lag.max=36);
```

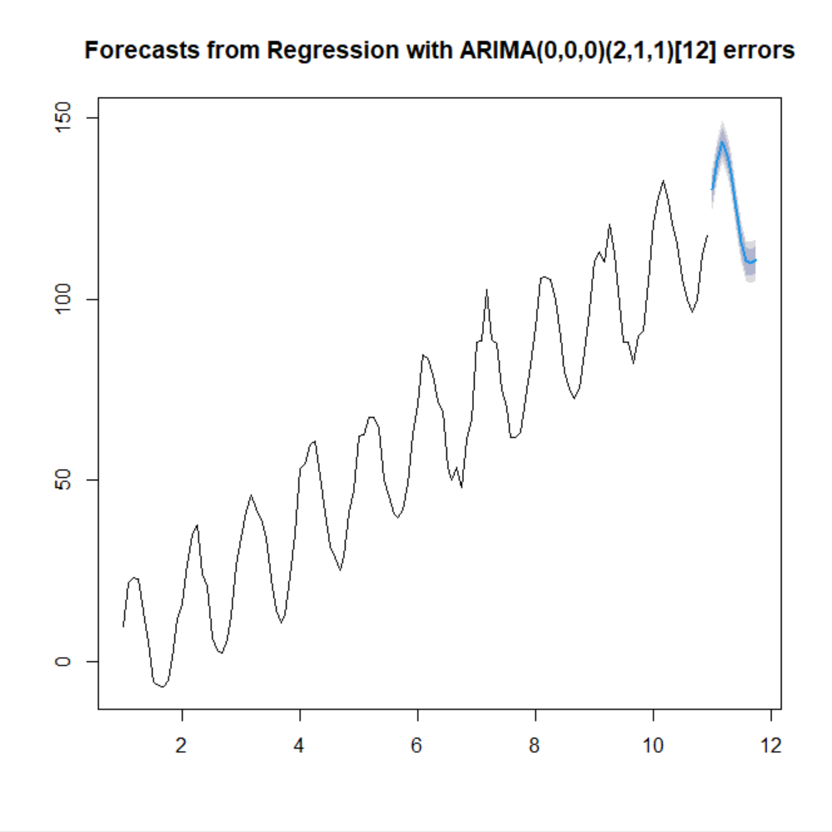
Multiplicative Seasonal ARIMA model

- Co2 data: Monthly Carbon Dioxide Levels at Alert, NWT, Canada

```
# 1st and seasonal difference of co2
x11(); plot(diff(diff(co2), lag=12), ylab='1st and seasonal difference of co2', xlab='time');
x11(); pacf(as.vector(diff(diff(co2), lag=12)), lag.max=36, ci.type='ma');

# estimation of ARIMA(1,1,1)(1,1,1)12
auto.arima(co2);   # Best model fitting
M1.co2=arima(co2, order=c(1,1,1), seasonal=list(order=c(1,1,2), period=12)); M1.co2;
f1=forecast(M1.co2, h=30);
accuracy(f1);

# residual analysis
x11(); plot(window(rstandard(M1.co2), start=c(1995,2)), ylab='standardized residuals', type='o');
abline(h=0);

x11(); acf(as.vector(window(rstandard(M1.co2), start=c(1995,2))), lag.max=36);

x11(); win.graph(width=3, height=3, pointsize=8);
hist(window(rstandard(M1.co2), start=c(1995, 2)), xlab= 'standardized residuals');

x11(); win.graph(width=3, height=3, pointsize=8);
qqnorm(window(rstandard(M1.co2), start=c(1995,2));
qqline(window(rstandard(M1.co2), start=c(1995,2));
```

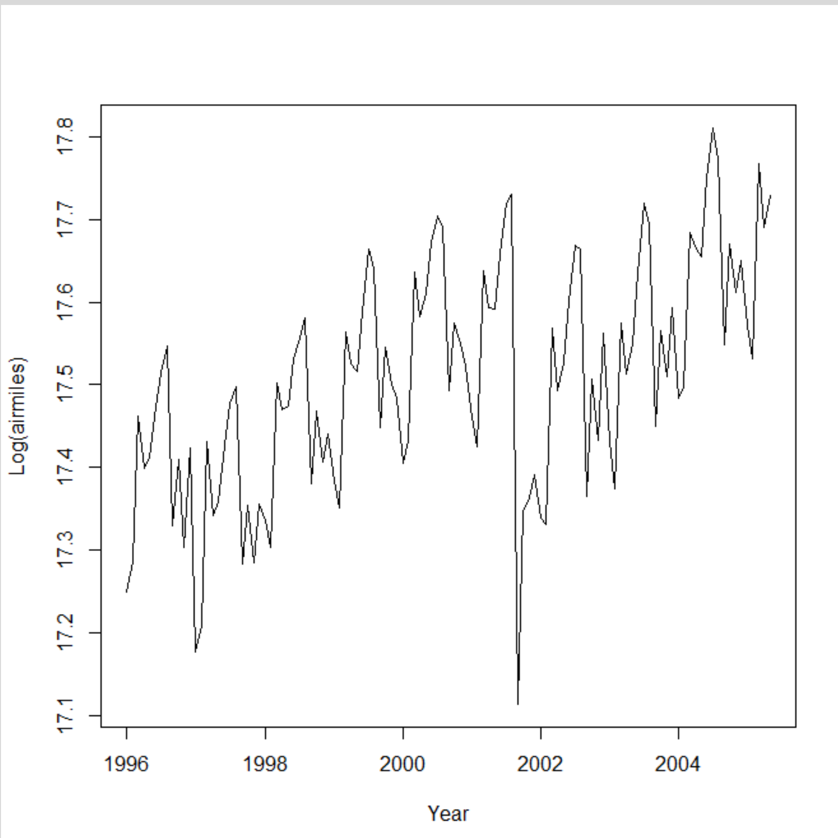
ARIMAX

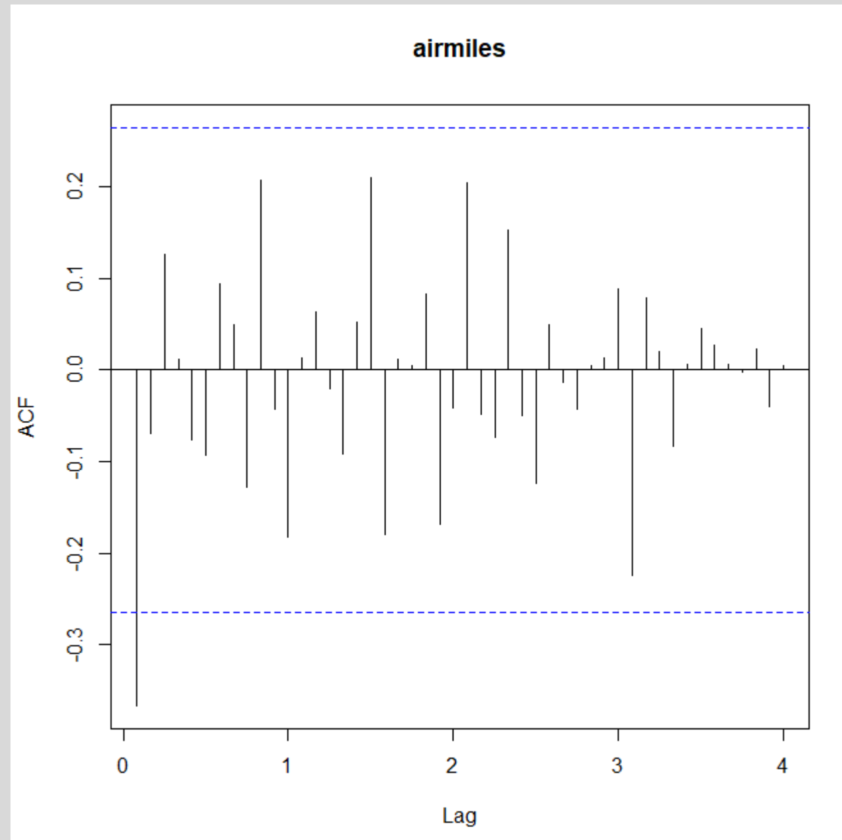
```
library(forecast);
# simulate random data
x <- ts(rnorm(120,0,3) + 1:120 + 20*sin(2*pi*(1:120)/12), frequency=12);
temp = rnorm(length(x), 20, 30); # exogenous variable

# build the ARMAX model
model1 = auto.arima(x, xreg = temp);

# model summary
summary(model1);

# new predictors: 10 data points
temp.reg = rnorm(10, 20, 30);
n=length(temp.reg);
x11(); plot(1:n, temp.reg, type='l');

# forecasting
forecast1 = forecast(model1, xreg = temp.reg);

# visualize
x11(); plot(forecast1);

# model info
summary(forecast1);
```



ARIMAX

```
library(TSA);
data(airmiles);
x11(); plot(log(airmiles),ylab='Log(airmiles)',xlab='Year', main='') ;
x11(); plot(acf(diff(diff(window(log(airmiles), end=c(2001,8)), 12)), lag.max=48,main=''));
```





ARIMAX

```
# in-sample & out-of-sample
airmiles_in=airmiles[1:100,];  # 1996.1~2004.4
reg=c(rep(0,11),1,1,rep(0,87))  # dummy regress: 100 points
airmiles_out=airmiles[101:113,]; # 2004.5~2005.5

# build the ARMAX model using in-sample
model2=auto.arima(log(airmiles_in), xreg=reg);

# model summary
summary(model2);

# new predictors: new dummy regress: 13 points
new.reg=c(rep(0, 7), 1, rep(0, 5));

# forecasting
forecast2 = forecast(model2, xreg = new.reg);

# prediction accuracy
x11(); plot(forecast2);
pred=exp(forecast2$mean);
diff=(airmiles_out-pred)^2;
mse=mean(diff);
rmse=sqrt(mse);
rmse/mean(airmiles);
```