Variograms

2026 January 30

Required library

```
library(forecast)
# This is needed for
# forecast::auto.arima(y,d,seasonal,stationary)
# See Section 8.1 to 8.7 of H&A textbook
```

Functions to predict up to n.ahead, and get lower limit, point forecast, upper limit

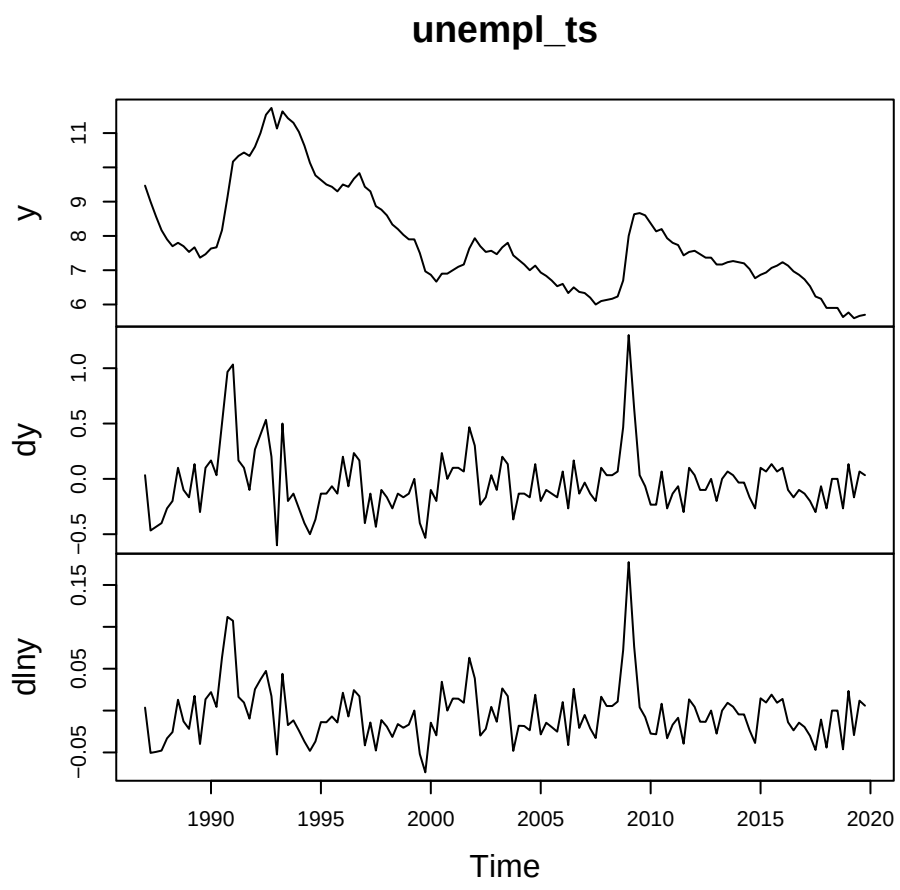
```
#' @description
#' Get 1 to n.ahead step forecasts at end of training set for dlny ( diff(log(y)) )
#' @param armaobj output object from arima or auto.arima
#' @param n.ahead number of steps ahead, for predict() function
#' @param y_ntrain last y value in training set
#' @param se.fit logical value for predict()
#' @param alpha  for central 100*(1-alpha)% prediction intervals;
#'     alpha=0.10 for 90% intervals
#' @return prediction interval for 1-step ahead forecast;
#'    vectors of point forecasts and corresponding SEs.
#' @details   1 to n.ahead step forecasts are printed for dlny, not y
dlny_predict = function(armaobj, y_ntrain, n.ahead, se.fit, alpha)
{ predobj = predict(armaobj, n.ahead=n.ahead, se.fit=se.fit)
  cv = qnorm(1-alpha/2)
  lwr = predobj$pred - cv*predobj$se
  upr = predobj$pred + cv*predobj$se
  out = cbind(lwr, predobj$pred ,upr)
  colnames(out) = c("lwr","pred","upr")
  print(out)
  # prediction interval for Y_{ntrain+1} : 1-step forecast
  ypred1step = exp(out[1,]) * y_ntrain
  cat("1-step ahead forecast interval for y\n")
  print(round(ypred1step,3))
  list(ypred1step=ypred1step, pred=predobj$pred, se=predobj$se)
}

#' @description
#' Get 1 to n.ahead step forecasts at end of training set for dy  ( diff(y) )
#' @param armaobj output object from arima or auto.arima
#' @param n.ahead number of steps ahead, for predict() function
#' @param y_ntrain last y value in training set
#' @param se.fit logical value for predict()
```

```
#' @param alpha  for central 100*(1-alpha)% prediction intervals;
#'     alpha=0.10 for 90% intervals
#' @return prediction interval for 1-step ahead forecast;
#'    vectors of point forecasts and corresponding SEs.
#' @details   1 to n.ahead step forecasts are printed for dy, not y
dy_predict = function(armaobj, y_ntrain, n.ahead, se.fit, alpha)
{ predobj = predict(armaobj, n.ahead=n.ahead, se.fit=se.fit)
  cv = qnorm(1-alpha/2)
  lwr = predobj$pred - cv*predobj$se
  upr = predobj$pred + cv*predobj$se
  out = cbind(lwr, predobj$pred ,upr)
  colnames(out) = c("lwr","pred","upr")
  print(out)
  # prediction interval for Y_{ntrain+1} : 1-step forecast
  ypred1step = out[1,]+y_ntrain
  cat("1-step ahead forecast interval for y\n")
  print(round(ypred1step,3))
  list(ypred1step=ypred1step, pred=predobj$pred, se=predobj$se)
}

#' @description
#' Get 1 to n.ahead step forecasts at end of training set for y
#' @param armaobj output object from arima or auto.arima
#' @param n.ahead number of steps ahead, for predict() function
#' @param se.fit logical value for predict()
#' @param alpha  for central 100*(1-alpha)% prediction intervals;
#'     alpha=0.10 for 90% intervals
#' @return prediction interval for 1-step ahead forecast;
#'    vectors of point forecasts and corresponding SEs.
#' @details   1 to n.ahead step forecasts are printed for y
y_predict = function(armaobj, n.ahead, se.fit, alpha)
{ predobj = predict(armaobj, n.ahead=n.ahead, se.fit=se.fit)
  cv = qnorm(1-alpha/2)
  lwr = predobj$pred - cv*predobj$se
  upr = predobj$pred + cv*predobj$se
  out = cbind(lwr, predobj$pred ,upr)
  colnames(out) = c("lwr","pred","upr")
  print(out)
  # prediction interval for Y_{ntrain+1} : 1-step forecast
  ypred1step = out[1,]
  cat("1-step ahead forecast interval for y\n")
  print(round(ypred1step,3))
  list(ypred1step=ypred1step, pred=predobj$pred, se=predobj$se)
}
```

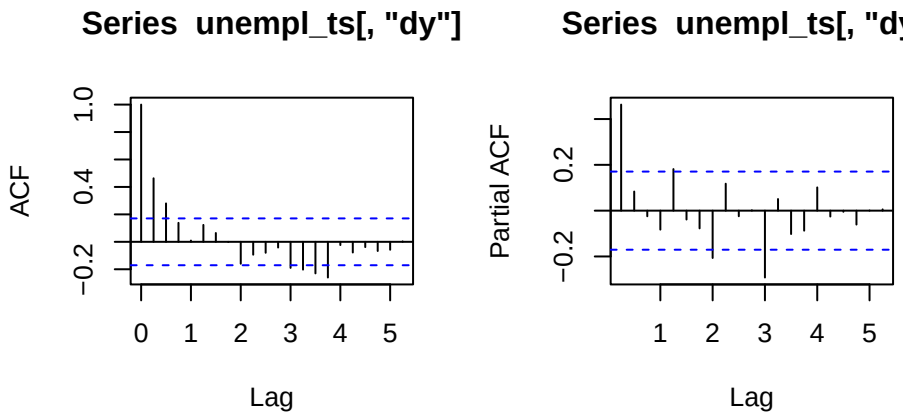
Set up dataframe after some wrangling and create some visualization plots

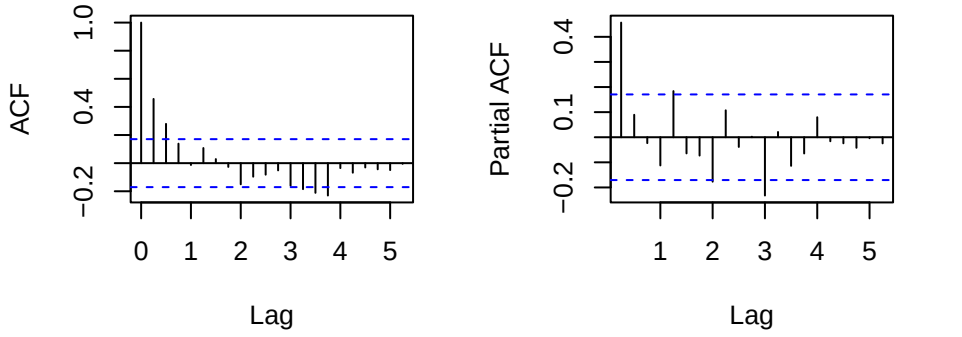
```
df = read.csv("gdp-unemploy-FRED.csv", header=T, nrow=236)
# Rename some variable, create other variables from unemployment

# Generic y variable is used in order that code can be used for other variables.
```

```r
y = df$unempl
dy = c(NA,diff(y))
dlny = df$diflnunempl/100
# Change from 100[log(y_t)-log_{y-1}] to [log(y_t)-log_{y-1}] for simpler
# interpretation later

unempl_df = data.frame(y=y,dy=dy,dlny=dlny)
# restrict to start in 1987 rather than 1961, omit 104 rows
unempl_df = unempl_df[-(1:104),]  # now 1987-01-01 to 2019-10-01 by quarter

# Create ts object
unempl_ts = ts(unempl_df, start=c(1987,1),end=c(2019,4),frequency=4)
plot(unempl_ts)
```




```r
par(mfrow=c(2,2))
acf(unempl_ts[,'dy']);  pacf(unempl_ts[,'dy'])
acf(unempl_ts[,'dlny']);  pacf(unempl_ts[,'dlny'])
```






```
# pacf for partial autocorrelation function for conditional dependence .
# To be explained later.

ntotal = nrow(unempl_ts)

ntrain = 80
ytrain = window(unempl_ts,start=c(1987,1),end=c(2006,4))
ntrain = nrow(ytrain)

summary(ytrain)
#>        y                dy                 dlny          
#>  Min.   : 6.333   Min.   :-0.60000   Min.   :-0.073766  
#>  1st Qu.: 7.350   1st Qu.:-0.20000   1st Qu.:-0.025303  
#>  Median : 7.900   Median :-0.10000   Median :-0.013525  
#>  Mean   : 8.462   Mean   :-0.03833   Mean   :-0.004915  
#>  3rd Qu.: 9.500   3rd Qu.: 0.13333   3rd Qu.: 0.016468  
#>  Max.   :11.733   Max.   : 1.03334   Max.   : 0.111870
```

Time series model for difference of consecutive log(unemployment rate)

```
# auto.arima tries to fit several time series models and returns a model
# that minimizes AIC = Akaike information criterion
# log-likelihood and AIC to be explained later.
# d= 0 for no differencing, stationary=T to find "best" ARMA model
dlny_auto = forecast::auto.arima(ytrain[,'dlny'],seasonal=F,stationary=T)
print(dlny_auto)
#> Series: ytrain[, "dlny"]
#> ARIMA(2,0,0) with zero mean
#>
#> Coefficients:
#>          ar1     ar2
#>       0.3603  0.1662
#> s.e.  0.1105  0.1120
#>
#> sigma^2 = 0.0009315:  log likelihood = 166.51
#> AIC=-327.03   AICc=-326.71   BIC=-319.88

# Check if stats::arima() has same results
# AR(2) for dlny with maximum likelihood (ML) estimation
dlny_ar2ml = arima(ytrain[,'dlny'],order=c(2,0,0),method="ML")
print(dlny_ar2ml)
#>
#> Call:
#> arima(x = ytrain[, "dlny"], order = c(2, 0, 0), method = "ML")
#>
#> Coefficients:
#>          ar1     ar2  intercept
#>       0.3532  0.1597    -0.0051
#> s.e.  0.1106  0.1120     0.0068
#>
#> sigma^2 estimated as 0.0009023:  log likelihood = 166.79,  aic = -325.57

# AR(2) for dlny with  conditional sum of squares (CSS) estimation
dlny_ar2css = arima(ytrain[,'dlny'],order=c(2,0,0),method="CSS")
print(dlny_ar2css)
#>
#> Call:
#> arima(x = ytrain[, "dlny"], order = c(2, 0, 0), method = "CSS")
#>
#> Coefficients:
#>          ar1     ar2  intercept
#>       0.3648  0.1582    -0.0039
#> s.e.  0.1092  0.1097     0.0070
#>
#> sigma^2 estimated as 0.0008941:  part log likelihood = 167.27

# AR(2) for dlny with using least squares and stats::lm()
# AR2 lm with dlny
dlny = ytrain[,'dlny']
```

```r
dlny_lm = lm(dlny[-c(1,2)] ~ dlny[-c(1,ntrain)]+ dlny[-c(ntrain-1,ntrain)])
print(summary(dlny_lm))
#>
#> Call:
#> lm(formula = dlny[-c(1, 2)] ~ dlny[-c(1, ntrain)] + dlny[-c(ntrain -
#>     1, ntrain)])
#>
#> Residuals:
#>       Min        1Q    Median        3Q       Max
#> -0.064382 -0.015163 -0.004459  0.021655  0.090006
#>
#> Coefficients:
#>                               Estimate Std. Error t value Pr(>|t|)
#> (Intercept)                  -0.001871   0.003504  -0.534  0.59487
#> dlny[-c(1, ntrain)]           0.364772   0.112754   3.235  0.00181 **
#> dlny[-c(ntrain - 1, ntrain)]  0.158226   0.113310   1.396  0.16672
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.03049 on 75 degrees of freedom
#> Multiple R-squared:  0.2124, Adjusted R-squared:  0.1914
#> F-statistic: 10.11 on 2 and 75 DF,  p-value: 0.0001294

y_ntrain = unempl_df$y[ntrain]
alpha = 0.10

pred1 = dlny_predict(dlny_auto, y_ntrain, n.ahead=4, se.fit=T, alpha)
#>                 lwr         pred        upr
#> 2007 Q1 -0.05335204 -0.003149727 0.04705258
#> 2007 Q2 -0.05794137 -0.004580013 0.04878134
#> 2007 Q3 -0.05756601 -0.002173717 0.05321857
#> 2007 Q4 -0.05756427 -0.001544490 0.05447529
#> 1-step ahead forecast interval for y
#>   lwr  pred   upr
#> 6.036 6.347 6.673
pred2 = dlny_predict(dlny_ar2ml, y_ntrain, n.ahead=4, se.fit=T, alpha)
#>                 lwr         pred        upr
#> 2007 Q1 -0.05504718 -0.005639112 0.04376895
#> 2007 Q2 -0.06016809 -0.007769369 0.04462935
#> 2007 Q3 -0.06036271 -0.006112258 0.04813819
#> 2007 Q4 -0.06066841 -0.005867239 0.04893393
#> 1-step ahead forecast interval for y
#>   lwr  pred   upr
#> 6.026 6.331 6.652
pred3 = dlny_predict(dlny_ar2css, y_ntrain, n.ahead=4, se.fit=T, alpha)
#>                 lwr         pred        upr
#> 2007 Q1 -0.05450574 -0.005322231 0.04386128
#> 2007 Q2 -0.05944544 -0.007091577 0.04526228
#> 2007 Q3 -0.05957864 -0.005300012 0.04897861
#> 2007 Q4 -0.05980086 -0.004926385 0.04994809
#> 1-step ahead forecast interval for y
#>   lwr  pred   upr
#> 6.029 6.333 6.652
```

```
# Compare parametrizations
b_css = dlny_ar2css$coef
print(b_css)  # from arima()
#>          ar1          ar2    intercept
#>  0.364793477  0.158207998 -0.003922509
phi1 = b_css[1]; phi2 = b_css[2]; mu = b_css[3]
b0_reg = mu*(1-phi1-phi2)
print(b0_reg) # this is same as intercept from lm
#>    intercept
#> -0.001871031
print(dlny_lm$coef) # from lm(); estimated slopes match
#>                   (Intercept)          dlny[-c(1, ntrain)]
#>                  -0.001871319                  0.364771857
#> dlny[-c(ntrain - 1, ntrain)]
#>                   0.158225512
```

Comparing lm() and arima when model is AR; this is illustrated for AR(2), but the pattern works for AR(p). The prediction equation for lm() is

$$\hat{\beta}_0 + \hat{\beta}_1 * dlny_{t-1} + \hat{\beta}_2 * dlny_{t-2}$$

The prediction equation for arima(order=c(2,0,0)) is

$$\hat{\mu} + \hat{\phi}_1 * (dlny_{t-1} - \hat{\mu}) + \hat{\phi}_2 * (dlny_{t-2} - \hat{\mu})$$

Hence

$$\hat{\beta}_1 = \hat{\phi}_1$$
$$\hat{\beta}_2 = \hat{\phi}_2$$
$$\hat{\beta}_0 = \hat{\mu}(1 - \hat{\phi}_1 - \hat{\phi}_2)$$

This explains the above match of output for arima() and lm().

Time series model for difference of consecutive unemployment rate

```
# Response variable is dy ; ARIMA(p,0,q) is used
# Response variable is now dy = diff(y) and not diff(log(y))
dy_auto = auto.arima(ytrain[,'dy'],seasonal=F,stationary=T)
print(dy_auto)
#> Series: ytrain[, "dy"]
#> ARIMA(1,0,0) with zero mean
#>
#> Coefficients:
#>          ar1
#>       0.4387
#> s.e.  0.0994
#>
#> sigma^2 = 0.07318:  log likelihood = -8.53
#> AIC=21.05   AICc=21.21   BIC=25.82

y_ntrain = unempl_df$y[ntrain]
alpha = 0.10
```

```
pred4 = dy_predict(dy_auto, y_ntrain, n.ahead=4, se.fit=T, alpha)
#>                lwr        pred       upr
#> 2007 Q1 -0.5034703 -0.05849766 0.3864750
#> 2007 Q2 -0.5115797 -0.02566488 0.4602500
#> 2007 Q3 -0.5046660 -0.01126005 0.4821459
#> 2007 Q4 -0.4997750 -0.00494016 0.4898947
#> 1-step ahead forecast interval for y
#>   lwr  pred   upr
#> 5.863 6.308 6.753

# AR(1) for dy with maximum likelihood (ML) estimation, compare auto.arima
dy_ar1 = arima(ytrain[,'dy'],order=c(1,0,0), method="ML")
print(dy_ar1)
#>
#> Call:
#> arima(x = ytrain[, "dy"], order = c(1, 0, 0), method = "ML")
#>
#> Coefficients:
#>          ar1  intercept
#>       0.4297    -0.0385
#> s.e.  0.0999     0.0520
#>
#> sigma^2 estimated as 0.07179:  log likelihood = -8.26,  aic = 22.52
pred5 = dy_predict(dy_ar1, y_ntrain, n.ahead=4, se.fit=T, alpha)
#>                lwr        pred       upr
#> 2007 Q1 -0.5199880 -0.07926439 0.3614592
#> 2007 Q2 -0.5357146 -0.05603369 0.4236472
#> 2007 Q3 -0.5325840 -0.04605257 0.4404789
#> 2007 Q4 -0.5295497 -0.04176415 0.4460214
#> 1-step ahead forecast interval for y
#>   lwr  pred   upr
#> 5.847 6.287 6.728
```

Time series model for unemployment rate

```
# Response variable is y ; ARIMA(p,1,q) is used
y_auto = auto.arima(ytrain[,'y'],d=1,seasonal=F,stationary=F)
print(y_auto)
#> Series: ytrain[, "y"]
#> ARIMA(3,1,1)
#>
#> Coefficients:
#>           ar1     ar2     ar3     ma1
#>       -0.5221  0.4675  0.2438  0.9491
#> s.e.   0.1384  0.1256  0.1172  0.1059
#>
#> sigma^2 = 0.06949:  log likelihood = -5.18
#> AIC=20.35   AICc=21.18   BIC=32.2

# ARIMA(3,1,1) for y with maximum likelihood (ML) estimation, compare auto.arima
y_arima311 = arima(ytrain[,'y'],order=c(3,1,1))
print(y_arima311)
```

```
#>
#> Call:
#> arima(x = ytrain[, "y"], order = c(3, 1, 1))
#>
#> Coefficients:
#>           ar1     ar2     ar3     ma1
#>       -0.5221  0.4675  0.2438  0.9491
#> s.e.   0.1384  0.1256  0.1172  0.1059
#>
#> sigma^2 estimated as 0.06597:  log likelihood = -5.18,  aic = 20.35

# Note the inconsistency in auto.arima with the original and differenced series.
# This is likely because several different models have similar AIC values.

# Note that auto.arima and arima agree for parameter estimates, but have
# small differences (second decimal place) for forecast intervals

# Up to 4-step ahead forecast at end of training set
alpha = 0.10
pred6 = y_predict(y_auto, n.ahead=4, se.fit=T, alpha)
#>              lwr     pred      upr
#> 2007 Q1 5.841019 6.274612 6.708206
#> 2007 Q2 5.545464 6.300975 7.056485
#> 2007 Q3 5.164733 6.211668 7.258604
#> 2007 Q4 4.892275 6.248180 7.604086
#> 1-step ahead forecast interval for y
#>   lwr  pred   upr
#> 5.841 6.275 6.708

pred7 = y_predict(y_arima311, n.ahead=4, se.fit=T, alpha)
#>              lwr     pred      upr
#> 2007 Q1 5.852142 6.274612 6.697083
#> 2007 Q2 5.564845 6.300975 7.037104
#> 2007 Q3 5.191591 6.211668 7.231746
#> 2007 Q4 4.927059 6.248180 7.569302
#> 1-step ahead forecast interval for y
#>   lwr  pred   upr
#> 5.852 6.275 6.697


# ARIMA(1,1,0) for y with maximum likelihood (ML) estimation,
# compare ARMA(1,0,0) for dy = differenced series
# A difference is due to no intercept term (mu=0) for ARIMA;
#    the AR1 coefficent and sigma^2 estimates are qualitatively similar.
y_arima110 = arima(ytrain[,'y'],order=c(1,1,0))
print(y_arima110)
#>
#> Call:
#> arima(x = ytrain[, "y"], order = c(1, 1, 0))
#>
#> Coefficients:
#>          ar1
#>       0.4544
```

```
#> s.e.  0.1010
#>
#> sigma^2 estimated as 0.07243:  log likelihood = -8.52,  aic = 21.04
```

1-step forecast interval at end of training set

```
oneahead = rbind(pred1$ypred1step,pred2$ypred1step,pred3$ypred1step,
   pred4$ypred1step,pred5$ypred1step,pred6$ypred1step,pred7$ypred1step)
# 7 comparisons: auto.arima, arima(ML), arima(CSS) for dlny;
# auto.arima, arima(ML) for dy;       auto.arima, arima(ML) for y.
rownames(oneahead) = c("dlny1", "dlny2", "dlny3", "dy1", "dy2", "y1","y2")
print(oneahead)
#>             lwr     pred      upr
#> dlny1 6.035894 6.346645 6.673395
#> dlny2 6.025671 6.330866 6.651518
#> dlny3 6.028935 6.332872 6.652132
#> dy1   5.863197 6.308169 6.753142
#> dy2   5.846679 6.287403 6.728126
#> y1    5.841019 6.274612 6.708206
#> y2    5.852142 6.274612 6.697083

actual = unempl_df$y[ntrain+1]
print(actual)
#> [1] 6.333333
```

Many-step ahead point forecasts at end of training set

```
# Get 1-step to 4-step ahead forecasts
# from dlny model with pred2, dy2 model with pred5 and y model with pred7
y_ntrain = unempl_df$y[ntrain]
fc7 = pred7$pred

fc5 = pred5$pred
fc5 = y_ntrain + c( fc5[1], sum(fc5[1:2]), sum(fc5[1:3]), sum(fc5[1:4]) )

fc2 = pred2$pred
fc2 = y_ntrain * c( exp(fc2[1]), exp(sum(fc2[1:2])),
  exp(sum(fc2[1:3])), exp(sum(fc2[1:4])) )

print(fc5)
#> [1] 6.287403 6.231369 6.185316 6.143552
print(fc2)
#> [1] 6.330866 6.281869 6.243590 6.207065
print(fc7)
#>          Qtr1     Qtr2     Qtr3     Qtr4
#> 2007 6.274612 6.300975 6.211668 6.248180
```

Explanations for the point forecasts. Let $n = n_{train}$ be size of training set. Models with dy use variable y' for consecutive differences. So

$$\hat{y}_{n+2} = \hat{y}_{n+2|n} = y_n + \hat{y'}_{n+1} + \hat{y'}_{n+2}$$

$$\hat{y}_{n+3} = \hat{y}_{n+2|n} = y_n + \hat{y'}_{n+1} + \hat{y'}_{n+2} + \hat{y'}_{n+3}$$

Models with dlny use variable ℓ' for consecutive log differences or log of ratio of consecutive values. $\exp\{\hat{\ell'}_{n+h}\}$ estimates a ratio of consecutive values.

$$\hat{y}_{n+1} = \hat{y}_{n+1|n} = y_n * \exp\{\hat{\ell'}_{n+1}\}$$

$$\hat{y}_{n+2} = \hat{y}_{n+2|n} = y_n * \exp\{\hat{\ell'}_{n+1}\} * \exp\{\hat{\ell'}_{n+2}\} = y_n * \exp\{\hat{\ell'}_{n+1} + \hat{\ell'}_{n+2}\}$$

$$\hat{y}_{n+3} = \hat{y}_{n+3|n} = y_n * \exp\{\hat{\ell'}_{n+1} + \hat{\ell'}_{n+2} + \hat{\ell'}_{n+2}\}$$

Future topics

```
#
# To get 2-step ahead prediction intervals for y after fitting model for
# dy or dlny is non-trivial
# (need more theoretical calculations based on the theory).

# To be presented in future lectures and data examples.

# For ML, the likelihood function assumes epsilon's are iid N(0,sigma^2).
# For CSS, the assumption is that E(epsilon)=0, Var(epsilon)=sigma^2.
# Derivations of SEs for prediction intervals.
# Explanations for all outputs of arima().
# Moving 1-step forecasts for holdout set.

# Diagnostics for fitted model, e.g., compare sample acf versus model-based acf
# for (differenced) series, look at out-of-sample predictions.
# General statistical modelling: best model based on likelihood AIC need not
# be an adequate model if the classes of models being considered are too narrow.

# Advantage of differencing after log:
#   forecast intervals cannot have lower limit below 0?
```