Woolyarn-arma

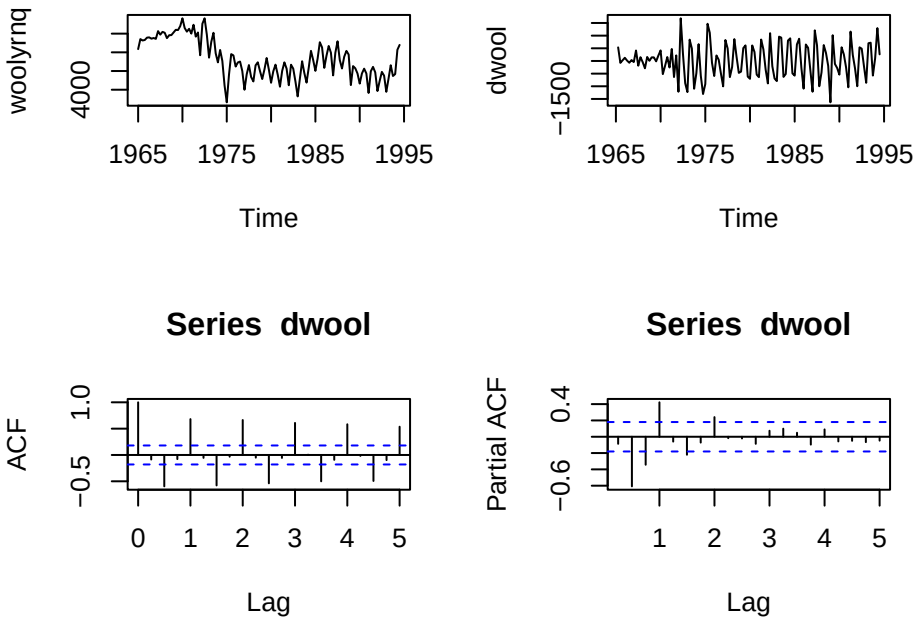
2026 February 04

Required library

```
library(forecast)
# woolyrnq: Quarterly production of woollen yarn in Australia: tonnes. Mar 1965 - Sep 1994
data(woolyrnq) # data set in library(forecast)
```

Set up training and holdout sets for differenced series

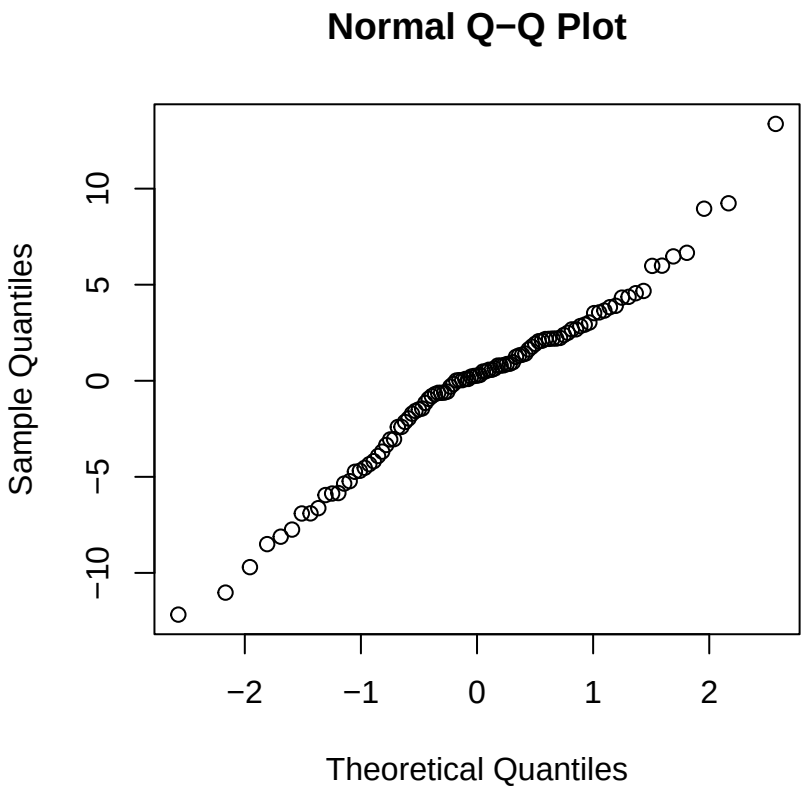
```
print(length(woolyrnq))
#> [1] 119
summary(woolyrnq)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>    3324    4882    5466    5658    6646    7819
par(mfrow=c(2,2))
dwool = diff(woolyrnq)
plot(woolyrnq); plot(dwool); acf(dwool)  # seasonal, lag 2 negative, lag 2 positive
pacf(dwool) # to be discussed later
```



```
summary(dwool)
#>      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
#> -1635.000  -518.000    10.000     1.898   517.000  1684.000

sd(dwool)
#> [1] 680.6576

dwool_tr = dwool/100  # transform to get smaller sigma estimate

ntotal = length(dwool_tr)
# Take first 100 as training set, or first 99 after differencing.
ntrain = 99 # missing first quarter so this is 25 years
train = dwool_tr[1:ntrain]
holdout = dwool_tr[(ntrain+1):ntotal] # 19 quarters or almost 5 years
```

ARMA models for diff(woolyrnq) – output used to demonstrate forecast theory

```
fit = auto.arima(train,seasonal=F,stationary=T)

qqnorm(fit$residuals)  # check normality assumption (for likelihood method)
```



```r
print(fit)
#> Series: train
#> ARIMA(3,0,3) with zero mean
#>
#> Coefficients:
#>           ar1      ar2      ar3     ma1     ma2     ma3
#>       -0.8558  -0.9653  -0.8395  0.6791  0.6927  0.4529
#> s.e.   0.2074   0.0320   0.2118  0.2904  0.1080  0.2978
#>
#> sigma^2 = 19.51:  log likelihood = -285.87
#> AIC=585.73   AICc=586.96   BIC=603.9

# Compare different ARMA models, verify auto.arima finds model with smallest AIC
fit_arma33 = arima(train,order=c(3,0,3),method="ML",include.mean=F)
fit_arma32 = arima(train,order=c(3,0,2),method="ML",include.mean=F)
fit_arma31 = arima(train,order=c(3,0,1),method="ML",include.mean=F)
fit_arma22 = arima(train,order=c(2,0,2),method="ML",include.mean=F)
fit_arma21 = arima(train,order=c(2,0,1),method="ML",include.mean=F)

print(fit_arma33) # difference in 3rd decimal place, compared with auto_arima
#>
#> Call:
#> arima(x = train, order = c(3, 0, 3), include.mean = F, method = "ML")
#>
#> Coefficients:
#>           ar1      ar2      ar3     ma1     ma2     ma3
#>       -0.8565  -0.9652  -0.8401  0.6802  0.6924  0.4537
#> s.e.   0.2078   0.0321   0.2122  0.2913  0.1081  0.2989
#>
#> sigma^2 estimated as 18.32:  log likelihood = -285.87,  aic = 585.73

print(fit_arma32)
#>
#> Call:
#> arima(x = train, order = c(3, 0, 2), include.mean = F, method = "ML")
#>
#> Coefficients:
#>           ar1      ar2      ar3     ma1     ma2
#>       -0.4299  -0.9587  -0.4094  0.1782  0.6558
#> s.e.   0.2401   0.0432   0.2014  0.2345  0.1617
#>
#> sigma^2 estimated as 18.91:  log likelihood = -287.24,  aic = 586.47

print(fit_arma31)
#>
#> Call:
#> arima(x = train, order = c(3, 0, 1), include.mean = F, method = "ML")
#>
#> Coefficients:
#>           ar1      ar2      ar3     ma1
#>       -0.7994  -0.7236  -0.6626  0.5101
#> s.e.   0.1198   0.0810   0.0861  0.1404
#>
```

```
#> sigma^2 estimated as 22.06:  log likelihood = -294.49,  aic = 598.99

print(fit_arma22)
#>
#> Call:
#> arima(x = train, order = c(2, 0, 2), include.mean = F, method = "ML")
#>
#> Coefficients:
#>          ar1      ar2      ma1     ma2
#>       0.0171  -0.9759  -0.0809  0.7639
#> s.e.  0.0247   0.0232   0.0830  0.0869
#>
#> sigma^2 estimated as 20.29:  log likelihood = -290.72,  aic = 591.44

# ARMA(2,1) will be used to match output of forecasts to theoretical results
print(fit_arma21)
#>
#> Call:
#> arima(x = train, order = c(2, 0, 1), include.mean = F, method = "ML")
#>
#> Coefficients:
#>          ar1      ar2      ma1
#>       0.0282  -0.6116  -0.2930
#> s.e.  0.1031   0.0796   0.1143
#>
#> sigma^2 estimated as 26.22:  log likelihood = -302.72,  aic = 613.43
```

1-step to 4-step forecasts at end of training set

```
pred_arma21 = predict(fit_arma21, n.ahead=4, se.fit=T); print(pred_arma21)
#> $pred
#> Time Series:
#> Start = 100
#> End = 103
#> Frequency = 1
#> [1]  0.4745704  1.7135010 -0.2419189 -1.0547196
#>
#> $se
#> Time Series:
#> Start = 100
#> End = 103
#> Frequency = 1
#> [1] 5.120159 5.296648 6.172528 6.216709

ar_coef = fit_arma21$coef[1:2]; ma_coef = fit_arma21$coef[3]
sigma = sqrt(fit_arma21$sigma2)
# Get the psi(b) coefficients
psiv = ARMAtoMA(ar=ar_coef, ma=ma_coef, lag.max=4)
mult = cumsum(c(1,psiv^2)); print(mult)
#> [1] 1.000000 1.070127 1.453313 1.474193 1.620605
print(sqrt(mult)*sigma) # Match the $se component of predict()
#> [1] 5.120159 5.296648 6.172528 6.216709 6.518115
```

```
# Code for the pi(b) coeffcients to come later
```

Moving 1-step ahead forecasts for holdout set?

```
# non-seasonal
#' @description
#' Moving 1-step ahead forecasts for holdout set
#' @param tsdata vector with training and holdout data
#' @param ntrain length of training set
#' @param order ARIMA order using for training set
#' @param method as used for arima applied to training set
#' @param traincoef the $coef component for arima applied to training set
#' @param includeman flag as used for arima applied to training set
#' @return holdout set rmse and 1-step forecasts
arma_fc = function(tsdata,ntrain,order,method,traincoef,include.mean, iprint=T)
{ obj = arima(tsdata,order=order,init=traincoef,fixed=traincoef,
     method=method, include.mean=include.mean,
     optim.control=list(maxit=0))  # no log-likelihood iterations
  # yt = y_{t|t-1} + innov ; fc = yt-innov
  fc = tsdata - obj$residuals
  # next line to compare with residuals when arima is applied only to training set
  #if(iprint) { print(round(obj$residuals,1)) }
  ntotal = length(tsdata)
  holdout_fc = fc[(ntrain+1):ntotal]
  holdout = tsdata[(ntrain+1):ntotal]
  if(iprint) print(cbind(holdout,holdout_fc))
  rmse = sqrt(mean((holdout-holdout_fc)^2))
  list(rmse=rmse, fc=holdout_fc)
}

arma21_fc = arma_fc(dwool_tr,99,c(2,0,1),method="ML",fit_arma21$coef,
  include.mean=F, iprint=T)
#>       holdout holdout_fc
#>  [1,]   -5.39  0.4745704
#>  [2,]    5.40  3.2665284
#>  [3,]    2.58  2.8233934
#>  [4,]   -2.59 -3.1583440
#>  [5,]  -10.41 -1.8173568
#>  [6,]   11.60  3.8081544
#>  [7,]    2.35  4.4102438
#>  [8,]   -2.94 -6.4241169
#>  [9,]   -9.98 -2.5409086
#> [10,]    5.39  3.6963222
#> [11,]    4.85  5.7590101
#> [12,]   -2.02 -2.8932025
#> [13,]   -8.64 -3.2788422
#> [14,]    7.00  2.5626197
#> [15,]    7.21  4.1809915
#> [16,]   -5.77 -4.9651400
#> [17,]    1.05 -4.3361465
#> [18,]   12.98  1.9800926
#> [19,]    2.61 -3.4992526
```

```r
print(arma21_fc$rmse)
#> [1] 5.107482

sd(fit_arma21$residuals)
#> [1] 5.137936

arma33_fc = arma_fc(dwool_tr,99,c(3,0,3),method="ML",fit_arma33$coef,
  include.mean=F, iprint=T)
#>       holdout holdout_fc
#>  [1,]   -5.39  -8.834729
#>  [2,]    5.40   7.414842
#>  [3,]    2.58   3.789510
#>  [4,]   -2.59  -3.548625
#>  [5,]  -10.41  -5.908161
#>  [6,]   11.60   6.301554
#>  [7,]    2.35   3.210668
#>  [8,]   -2.94  -3.423411
#>  [9,]   -9.98  -7.358872
#> [10,]    5.39   7.572881
#> [11,]    4.85   4.406212
#> [12,]   -2.02  -3.370903
#> [13,]   -8.64  -7.243829
#> [14,]    7.00   5.462455
#> [15,]    7.21   4.733262
#> [16,]   -5.77  -3.557663
#> [17,]    1.05  -6.990758
#> [18,]   12.98   3.674233
#> [19,]    2.61   3.609664

print(arma33_fc$rmse)
#> [1] 3.619187

sd(fit_arma33$residuals)
#> [1] 4.297631

# Results for arma_fc function to be confirmed with coding of the pi(b) function
```