ARIMA(p, d, q): ARIMA models represent a univariate time series as a combination of autoregressive (AR) terms, differencing for integration (I), and moving-average (MA) terms. An ARIMA(p, d, q) is built by: - differencing the series d times to achieve approximate stationarity; - modeling the differenced series with p autoregressive lags and q moving-average terms; - estimating parameters by maximum likelihood and selecting orders using an information criterion (e.g., AICc) and residual diagnostics.

In the example below, the function ts\_arima() performs automatic order selection and fitting. Forecasts are generated recursively for the desired number of steps ahead.

Objective: Fit and evaluate an ARIMA (AutoRegressive Integrated Moving Average) model for time-series forecasting, performing a train-test split, automatic order selection, and evaluation with metrics and visualization.

#install.packages("tspredit")  
  
# Loading the package  
library(tspredit)

# Series for study (no sliding window)  
  
data(tsd)  
ts <- ts\_data(tsd$y, 0)  
ts\_head(ts, 3)

## t0  
## [1,] 0.0000000  
## [2,] 0.2474040  
## [3,] 0.4794255

# Series visualization  
library(ggplot2)  
plot\_ts(x=tsd$x, y=tsd$y) + theme(text = element\_text(size=16))



# Train-test split and projection (X, y)  
  
samp <- ts\_sample(ts, test\_size = 5)  
io\_train <- ts\_projection(samp$train)  
io\_test <- ts\_projection(samp$test)

# Training the ARIMA model (orders selected automatically)  
  
model <- ts\_arima()  
model <- fit(model, x=io\_train$input, y=io\_train$output)

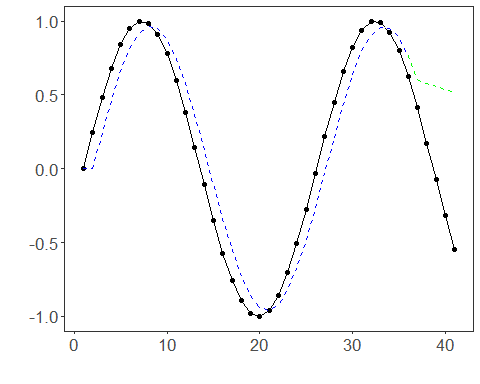
# Fit evaluation (train)  
  
adjust <- predict(model, io\_train$input)  
adjust <- as.vector(adjust)  
output <- as.vector(io\_train$output)  
ev\_adjust <- evaluate(model, output, adjust)  
ev\_adjust$mse

## [1] 0.02857686

# Forecast on test set (5 steps ahead)  
  
prediction <- predict(model, x=io\_test$input[1,], steps\_ahead=5)  
prediction <- as.vector(prediction)  
output <- as.vector(io\_test$output)  
ev\_test <- evaluate(model, output, prediction)  
ev\_test

## $values  
## [1] 0.41211849 0.17388949 -0.07515112 -0.31951919 -0.54402111  
##   
## $prediction  
## [1] 0.6011374 0.5784414 0.5566023 0.5355877 0.5153665  
##   
## $smape  
## [1] 1.489711  
##   
## $mse  
## [1] 0.4904025  
##   
## $R2  
## [1] -3.235632  
##   
## $metrics  
## mse smape R2  
## 1 0.4904025 1.489711 -3.235632

# Plot comparing actual vs fit (train) and forecast (test)  
  
yvalues <- c(io\_train$output, io\_test$output)  
plot\_ts\_pred(y=yvalues, yadj=adjust, ypre=prediction) + theme(text = element\_text(size=16))



References - G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung (2015). Time Series Analysis: Forecasting and Control. Wiley. - R. J. Hyndman and Y. Khandakar (2008). Automatic time series forecasting: The forecast package for R. Journal of Statistical Software, 27(3), 1–22. <doi:10.18637/jss.v027.i03>