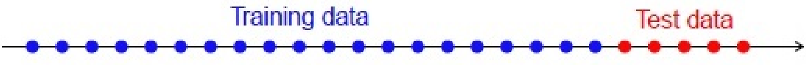
Applied Time Series Econometrics

Oxana Malakhovskaya, NRU HSE

October 29, 2019

Forecast evaluation

- Last time we made two (very similar) forecasts from two different models for one time series.
- Previously we have already compared candidate models in terms of their forecasting accuracy.
- To do it, we calculate pseudo real-time forecasts for the periods for which we have the data available and compute the measures of accuracy.



Series to compare

- For employment in manufacturing series:

```
empl_manuf <- read_csv("empl_manuf.csv")
empl <- ts(empl_manuf, start = c(1992, 1), freq = 12)
train = window(empl, end = c(2001,12))
fit1 <- Arima(train, order = c(3,2,4),
              seasonal = c(0,0,1))
fore1 <- forecast(fit1, h = 24)
fit2 <- auto.arima(train)
fore2 <- forecast(fit1, h = 24)
options(digits = 3)
accuracy(fore1, empl)
accuracy(fore2, empl)
```

Accuracy measures

```
##                     ME  RMSE   MAE       MPE  MAPE   MASE
## Training set -0.0149 0.818 0.498 -0.000115 0.362 0.0563
## Test set     -1.8850 2.470 2.002 -1.847460 1.953 0.2263
##                      ME RMSE   MAE      MPE  MAPE   MASE
## Training set -0.00506 0.86 0.504  0.00397 0.363 0.0569 -
## Test set     -1.94594 2.50 2.036 -1.90467 1.986 0.2301
```

Comparison with baseline alternatives

- Sometimes we are interested in comparing the forecast given by an econometric model with a simple benchmark.
- The main question here: is it reasonable to spend time on a relatively complicated model if a simple alternative has a similar forecasting performance?
- What are these useful simple alternatives that may play a role of a benchmark?

Simple forecasting models (1)

Average method

- A forecast of all future values is equal to mean of historic data $\{y_1, \dots, y_T\}$.
- Forecasts:

$$\hat{y}_{T+h|T} = \bar{y} = \frac{1}{T}\sum_{i=1}^{T} y_i$$

- It is a forecast using a WN with a constant model: $y_t = \mu + \nu_t$

Naïve method

- A forecast is equal to the last observed value.
- Forecasts:

$$\hat{y}_{T+h|T} = y_T$$

- It is a forecast using a random walk model: $y_t = y_{t-1} + \nu_t$

Simple forecasting models (2)

Seasonal naïve method

- A forecast is equal to the last value from the same season
- Forecasts:

$$\hat{y}_{T+h|T} = y_{T+h-km},$$

where m is a seasonal period and $k = \lfloor (h-1)/m \rfloor + 1$, where $\lfloor \cdot \rfloor$ means an integer part

- It is a forecast using a seasonal random walk model: $y_t = y_{t-m} + \nu_t$

Simple forecasting models(3)

Drift method

- A forecast is equal to the last value plus the average change
- Forecasts are:

$$y_{T+h,T} = y_T + \frac{h}{T-1}\sum_{t=2}^{T}(y_t - y_{t-1}) =$$
$$= y_T + \frac{h}{T-1}(y_T - y_1)$$

- Equvalent to extrapolating a line between the first and the last observations.
- It is a forecast using a random walk with drift model: $y_t = \mu + y_{t-1} + \nu_t$

Simple forecasts: codes

Mean forecast:

```
meanf(x, h = 12)
```

Naïve forecast

```
naive(x, h = 12)
rwf(x, h = 12)
```

Seasonal naïve forecast

```
snaive(x, h=12)
```

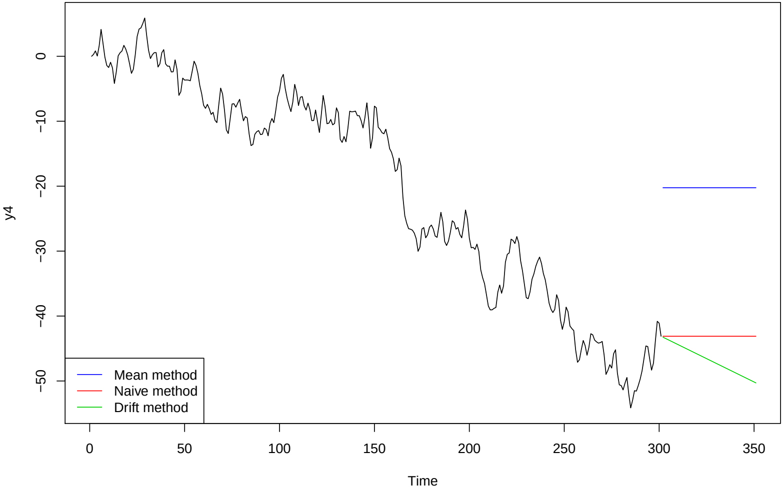
Drift forecast

```
rwf(x, drift = TRUE, h=12)
```

Example: code

```
set.seed(500)
y4 <- arima.sim(model = list(order = c(4,1,4),
ar = c(0.3, -0.4,0,-0.2), ma = c(0.8,0,0,0.5),
mean = 2), n = 300)
for1 <- meanf(y4, h = 50)
for2 <- naive(y4, h = 50)
for3 <- rwf(y4, drift = TRUE, h = 50)
plot(y4, xlim = c(1,350))
lines(for1$mean,col = 4)
lines(for2$mean, col = 2)
lines(for3$mean, col = 3)
legend("bottomleft",lty=1,col=c(4,2,3),
  legend=c("Mean method","Naive method","Drift method"))
```

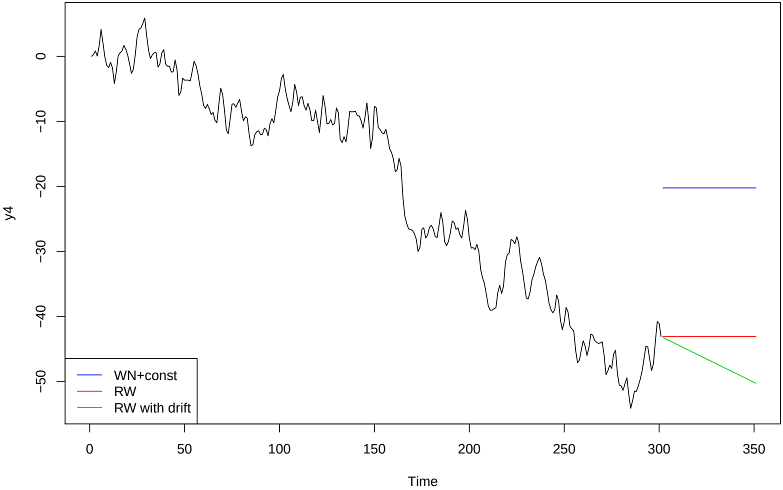
Simple forecasts: a graph



Sure, the same forecasts can be obtained by fitting an ARIMA model.

```
fit1 <- Arima(y4, order = c(0,0,0), include.constant = TRUE
fit2 <- Arima(y4, order = c(0,1,0))
fit3 <- Arima(y4, order = c(0,1,0), include.drift = TRUE)
fore1 <- forecast(fit1, h = 50)
fore2 <- forecast(fit2, h = 50)
fore3 <- forecast(fit3, h = 50)
plot(y4, xlim = c(1,350))
lines(fore1$mean, col = 4)
lines(fore2$mean, col = 2)
lines(fore3$mean, col = 3)
legend("bottomleft",lty=1,col=c(4,2,3),
       legend=c("WN+const","RW","RW with drift"))
```

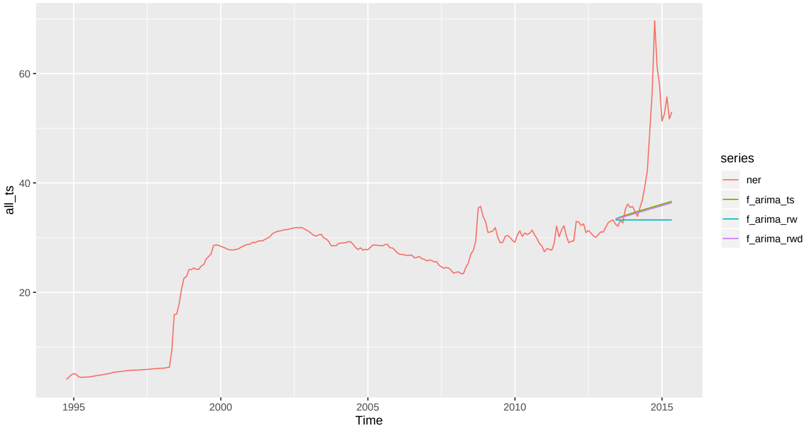
The same graph from the alternative code.



Application: a nominal exchange rate

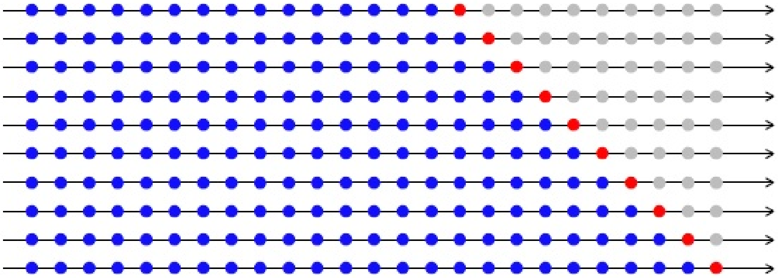
(see code)

```
autoplot(all_ts)
```



Cross-validation

- Cross-validation is very popular in time-series analysis as it allows the researcher not to loose some potentially important information.



Cross-validation algorithm

Assume k is the minimum number of observations for a training set.

- Select k, take $i = 0$ and use observations $1, 2, \ldots, k + i$ to estimate model. Compute error on forecast for time $k + i + h$ as made in period $k + i$.
- Repeat for $i = 1, \ldots, T - h - k$ where T is total number of observations.
- Compute accuracy measures over all errors for a given h.

Also called `rolling forecasting origin` because the origin $(k + i)$ at which forecast is based rolls forward in time.

Cross-validation example

We can do the cross validation with the `tsCV()` function from the forecast package.

```
# It is necessary to create a forecasting function first
fore_arima_212 <- function(y, h) {
  model <- Arima(y, order = c(2, 1, 2),
                 include.drift = TRUE)
  forecast <- forecast(model, h)
  return(forecast)
}
errors_cv <- tsCV(ner, fore_arima_212, h = 1)
errors_cv
```

NB! The forecasting errors are returned by the `tsCV()` function

Multivariate analysis: introduction

- A natural extension of the of the univariate ARMA class of models is one in which y_t represents a vector of n time series. Let Φ_i and Ψ_i be $(n \times n)$ matrices:

$$\Phi_i = \begin{pmatrix} \phi_{i,1,1} & \cdots & \phi_{i,1,n} \\ \vdots & \ddots & \vdots \\ \phi_{i,n,1} & \cdots & \phi_{i,n,n} \end{pmatrix} \qquad \Psi_i = \begin{pmatrix} \psi_{i,1,1} & \cdots & \psi_{i,1,n} \\ \vdots & \ddots & \vdots \\ \psi_{i,n,1} & \cdots & \psi_{i,n,n}, \end{pmatrix}$$

where $\phi_{i,j,k}$ is the autoregressive parameter at lag i in equation j on variable k and $\psi_{i,j,k}$ is the corresponding moving-average parameter.

Multivariate model classification(1)

- The multivariate analogue of the ARMA(p,q) model is the vector ARMA (VARMA(p,q)):

$$y_t = \mu + \sum_{i=1}^{p} \Phi_i y_{t-i} + \upsilon_t + \sum_{i=1}^{q} \Psi_i \upsilon_{t-i}, \qquad \upsilon_t \sim iidN(0, \Omega),$$

where υ_t is an n-dimensional disturbance vector with zero mean vector, μ is a n-dimensional vector, and $(n \times n)$ covariance matrix Ω and $\{\mu, \Phi_1, \ldots, \Phi_p, \Psi_1, \ldots, \Psi_q, \Omega\}$ are unknown parameters.

Multivariate model classification(2)

Using lag operators, the $VARMA(p,q)$ class of model is represented as:

$$\Phi_p(L)y_t = \mu + \Psi_q(L)\upsilon_t, \qquad \upsilon_t \sim iid(0,\Omega),$$

where:

$$\Phi_p(L) = I - \Phi_1 L - \Phi_2 L^2 - \cdots - \Phi_p L^p$$
$$\Psi_q(L) = I - \Psi_1 L - \Psi_2 L^2 - \cdots - \Psi_q L^p$$

are matrix polynomials in the lag operator L.

Vector autoregression (VAR)

An important special case of the VARMA model is one in which there are p autoregressive lags and no moving average lags, $q = 0$. This special case is a $VAR(p)$ model:

$$y_t = \mu + \Phi_1 y_{t-1} + \cdots + \Phi_p y_{t-p} + \upsilon_t$$

Properties

- Each variable is expressed as a function of its own lags and the lags of all the other variables in the system.
- The lag structure on all variables in all equations is the same.

! This is a reduced-form VAR. We will discuss a structural VAR models later.

VAR(1) model: example

A trivariate ($n = 3$) VAR model with one lag $VAR(1)$ is:

$$
\begin{aligned}
y_{1,t} &= \mu_1 + \phi_{1,1,1} y_{1,t-1} + \phi_{1,1,2} y_{2,t-1} + \phi_{1,1,3} y_{3,t-1} + \upsilon_{1,t} \\
y_{2,t} &= \mu_2 + \phi_{1,2,1} y_{1,t-1} + \phi_{1,2,2} y_{2,t-1} + \phi_{1,2,3} y_{3,t-1} + \upsilon_{2,t} \\
y_{3,t} &= \mu_3 + \phi_{1,3,1} y_{1,t-1} + \phi_{1,3,2} y_{2,t-1} + \phi_{1,3,3} y_{3,t-1} + \upsilon_{3,t}
\end{aligned}
$$

Example: matrix notation

In matrix notation, the model becomes:

$$\begin{pmatrix} y_{1,t} \\ y_{2,t} \\ y_{3,t} \end{pmatrix} = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{pmatrix} + \begin{pmatrix} \phi_{1,1,1} & \phi_{1,1,2} & \phi_{1,1,3} \\ \phi_{1,2,1} & \phi_{1,2,2} & \phi_{1,2,3} \\ \phi_{1,3,1} & \phi_{1,3,2} & \phi_{1,3,3} \end{pmatrix} \begin{pmatrix} y_{1,t-1} \\ y_{2,t-1} \\ y_{3,t-1} \end{pmatrix} + \begin{pmatrix} \upsilon_{1,t} \\ \upsilon_{2,t} \\ \upsilon_{3,t} \end{pmatrix}$$

or, more compactly,

$$y_t = \mu + \Phi_1 y_{t-1} + \upsilon_t$$

The model contains:

- 3 intercepts
- 9 autoregressive parameters
- 6 parameters in the covariance matrix

Stability condition

A formal statement of stability The n-dimensional variable y_t is stable provided that the roots of the polinomial $|\Phi_p(z)| = 0$ lie outside the unit circle ($|\cdot|$ denotes the determinant of the matrix).

$$y_t = \mu + \Phi_1 y_{t-1} + \Phi_2 y_{t-2} + \upsilon_t$$

Example:

$$\Phi_1 = \begin{pmatrix} 1.279 & -0.355 \\ 0.002 & 1.234 \end{pmatrix} \qquad \Phi_2 = \begin{pmatrix} -0.296 & 0.353 \\ 0.007 & -0.244 \end{pmatrix}$$

The polynomial $|I - \Phi_1 z - \Phi_2 z^2| = 0$

$$\left| \begin{pmatrix} 1 - 1.279z + 0.296z^2 & 0.355z - 0.353z^2 \\ -0.002z - 0.007z^2 & 1 + 1.234z + 0.244z^2 \end{pmatrix} \right| = 0$$

Solution

The equation may be written as:

$$(1 - 1.279z + 0.296z^2)(1 + 1.234z + 0.244z^2) - \\ - (0.355z - 0.353z^2)(-0.002z - 0.007z^2) = 0$$

The roots are:

$$z_1 = 4.757 \qquad z_2 = 2.874$$
$$z_3 = 1.036 \qquad z_2 = 1.011$$

Because $|z_i| > 1 \quad \forall i$ both processes are jointly stable.

Estimation

- The parameters of VARMA (exactly as ARMA) models are estimated by maximum likelihood method.
- For the important special case where there are no moving average terms, it is shown that the ML estimates are obtained by OLS.
- In R, we estimate VAR(p) models with the `VAR()` function from the `vars` package.
- But there are two decisions to make before using the software:

```
* How many variables to include (k)?
* What should be the order of the model (p)?
```

How many variables? Be reasonable!

- The number of coefficients to be estimated in VAR per equation is $1 + pn$, so the total number of parameters to estimate is:

$$\text{Number of parameters } = n(1 + pn)$$

  Example: $n = 5$, $p = 3$ so the total number of parameters to estimate is equal to 80.

The more coefficients to be estimated the larger the estimation error entering the forecast.

Conclusion: In practice it is usual to keep n small and include only variables that are correlated to each other and therefore useful in forecasting each other.

Literature

1. Martin, V., Hurn, S., and HarriS,D. (2013) Econometric Modelling with Time Series: Specification, Estimating and Testing, New York, Cambridge University Press
2. Hyndman, R. J. and Athanasopoulos, G. (2013) Forecasting: principles and practice, Otexts
3. Lütkepohl, H., and Krätzig, M. (2004) Applied Time Series Econometrics, New York, Cambridge University Press