Time Series Analysis

L. Torgo

ltorgo@fc.up.pt
Faculty of Sciences, University of Porto

May 2014

Time Series Data

A Definition

Definition

- A time series is a set of observations of a variable that are ordered by time.
- E.g.,
  $x_1, x_2, \cdots, x_{t-1}, x_t, x_{t+1}, \cdots, x_n$
  where x_t is the observation of variable X at time t.
- A multivariate time series is a set of observations of a set of variables over a certain period of time.

The Main Goals of Time Series Analysis

Explanation

Obtaining a Time Series Model help us to have
a Deeper Understanding of the Mechanism
that Generated the Observed Time Series Data.

Forecasting

- Given:
  $x_1, x_2, \cdots, x_{t-1}, x_t$ *The Past!*
- Obtain:
  a time series model
- Which is able to make predictions concerning:
  $x_{t+1}, \cdots, x_n$ *The Future!*

Other Goals

Time Series Data Mining

Main Time Series Data Mining Tasks

- *Indexing (Query by Content)*
  Given a query time series Q and a similarity measure $D(Q, X)$ find the most similar time series in a database **D**
- *Clustering*
  Find the natural goupings of a set of time series in a database **D** using some similarity measure $D(Q, X)$
- *Classification*
  Given an unlabelled time series Q, assign it a label C from a set of pre-defined labels (classes)

Time Series Data in R

- R has several data structures capable of handling time series data
- In our illustration we will use the infra-structure provided by package `xts`

```
> library(xts)
> data(ice.river,package='tseries')
> ice.river[1:4,]
     flow.vat flow.jok prec temp
[1,]     16.1     30.2  8.1  0.9
[2,]     19.2     29.0  4.4  1.6
[3,]     14.5     28.4  7.0  0.1
[4,]     11.0     27.8  0.0  0.6
> ir <- xts(ice.river[,1],
+           seq.Date(as.Date('1972-01-01'),
+                    by='day',
+                    len=nrow(ice.river)))
```


Time Series Data in R - indexing examples

```
> ir[1:3]
           [,1]
1972-01-01 16.1
1972-01-02 19.2
1972-01-03 14.5
> ir['1973-05-02']
           [,1]
1973-05-02 11.6
> ir['1972-01']
            [,1]
1972-01-01 16.10
1972-01-02 19.20
...
> ir['1972-01-23/1972-02-02']
           [,1]
1972-01-23 6.90
1972-01-24 6.90
...
```

```
> ir['/1972-01-10']
            [,1]
1972-01-01 16.10
1972-01-02 19.20
...
> ir['1974-12-21/']
           [,1]
...
1974-12-30 5.34
1974-12-31 5.34
> ir['1972-01-23/1972-02']
            [,1]
1972-01-23  6.90
1972-01-24  6.90
...
1974-12-30  5.34
1974-12-31  5.34
```

Summaries of Time Series Data

- Standard descriptive statistics (mean, standard deviation, etc.) do not allways work with time series (TS) data.
- TS may contain trends, seasonality and some other systematic components, making these stats misleading.
- So, for proving summaries of TS data we will be interested in concepts like trend, seasonality and correlation between sucessive observations of the TS.

Types of Variation

Seasonal Variation

Some time series exhibit a variation that is annual in period, e.g. demand for ice cream.

Other Cyclic Variation

Some time series have periodic variations that are not related to seasons but to other factors, e.g. some economic time series.

Trends

A trend is a long-term change in the mean level of the time series.

Stationarity

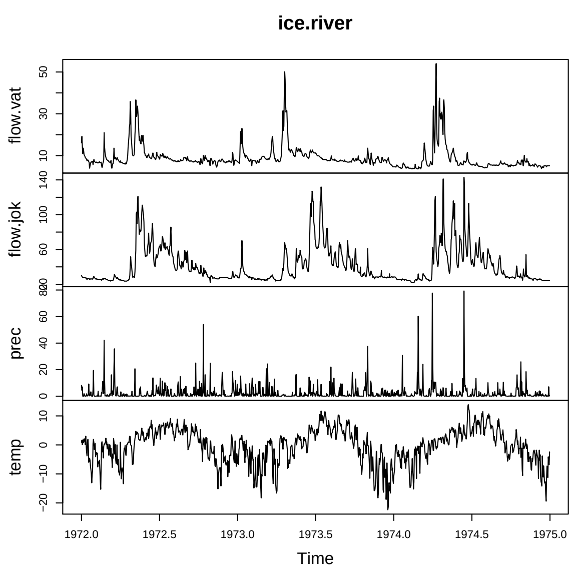
An Informal Definition

A time series is said to be stationary if

- there is no systematic change in mean (no trend),
- if there is no systematic change in variance and
- if strictly periodic variations have been removed.

Note that in these cases statistics like mean, standard deviation, variance, etc., bring relevant information!

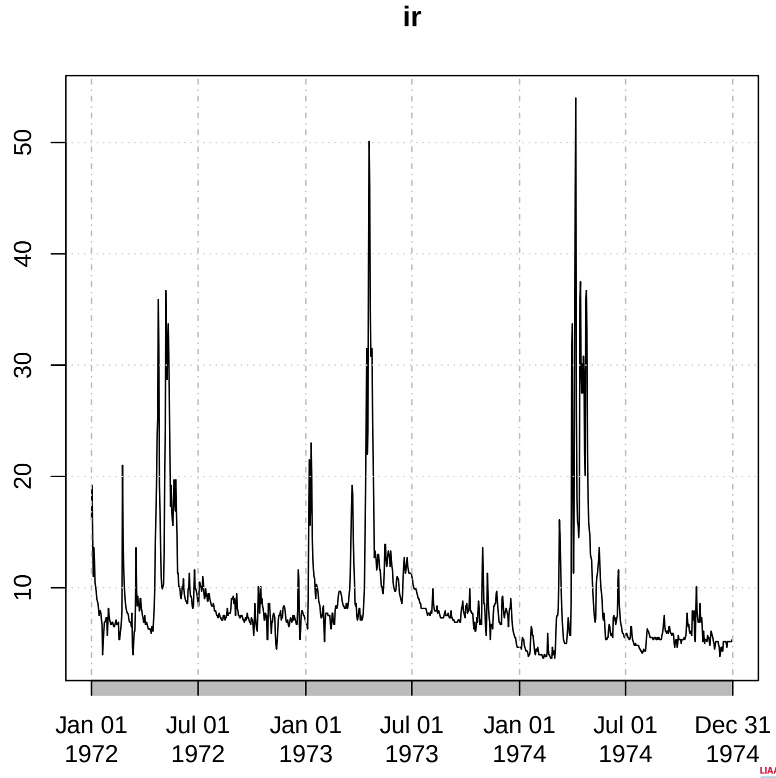
Time Plots



- Ploting the time series values against time is one of the most important tools for analysing its behaviour.
- Time plots show important features like trends, seasonality, outliers and discontinuities.

Time Plots in R

```
> plot(ir)
```



Transformations - I

Plotting the data may suggest transformations :

To stabilize the variance

Symptoms: trend with the variance increasing with the mean.
Solution: logarithmic transformation.

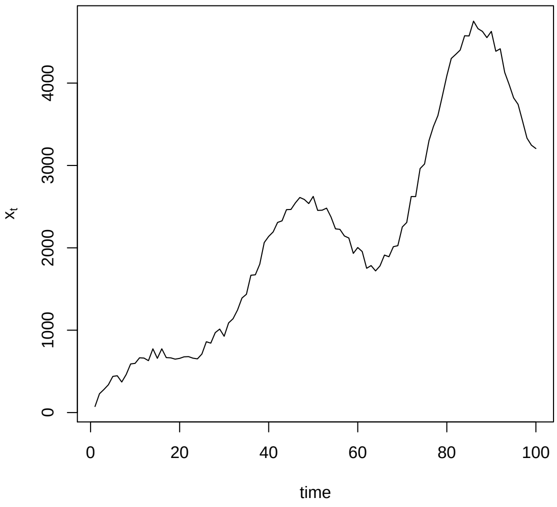
To make the seasonal effects additive

Symptoms: there is a trend and the size of the seasonal effect increases with the mean(multiplicative seasonality).
Solution: logarithmic transformation.

To remove trend

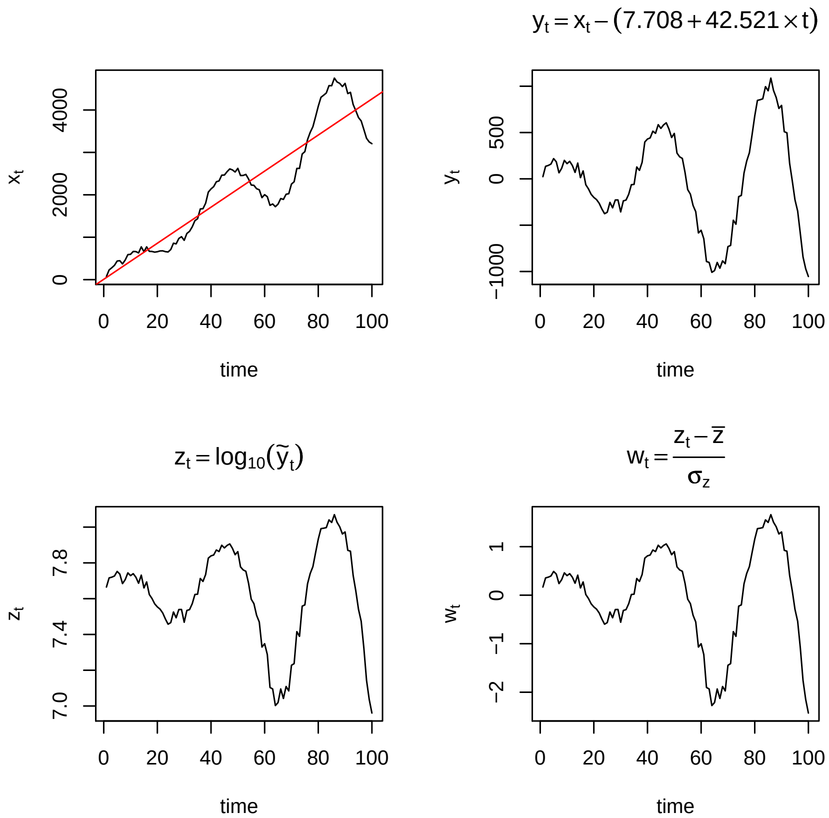
Symptoms: there is systematic change on the mean.
Solution 1: first order differentiation ($\nabla X_t = X_t - X_{t-1}$).
Solution 2: model the trend and subtractit from the original series ($Y_t = X_t - r_t$).

Transformations - a simple example (1)



An example time series with trend and a multiplicative seasonality effect.

Transformations - a simple example (2)



Some useful functions in R

```
> (s <- ir[1:10])
            [,1]
1972-01-01 16.10
1972-01-02 19.20
1972-01-03 14.50
1972-01-04 11.00
1972-01-05 13.60
1972-01-06 12.50
1972-01-07 10.50
1972-01-08 10.10
1972-01-09  9.68
1972-01-10  9.02
> diff(s)
            [,1]
1972-01-01    NA
1972-01-02  3.10
1972-01-03 -4.70
1972-01-04 -3.50
1972-01-05  2.60
1972-01-06 -1.10
1972-01-07 -2.00
1972-01-08 -0.40
1972-01-09 -0.42
1972-01-10 -0.66
```

```
> diff(s,diff=2)
            [,1]
1972-01-01    NA
1972-01-02    NA
1972-01-03 -7.80
1972-01-04  1.20
1972-01-05  6.10
1972-01-06 -3.70
1972-01-07 -0.90
1972-01-08  1.60
1972-01-09 -0.02
1972-01-10 -0.24
> log10(s)
                [,1]
1972-01-01 1.2068259
1972-01-02 1.2833012
1972-01-03 1.1613680
1972-01-04 1.0413927
1972-01-05 1.1335389
1972-01-06 1.0969100
1972-01-07 1.0211893
1972-01-08 1.0043214
1972-01-09 0.9858754
1972-01-10 0.9552065
```

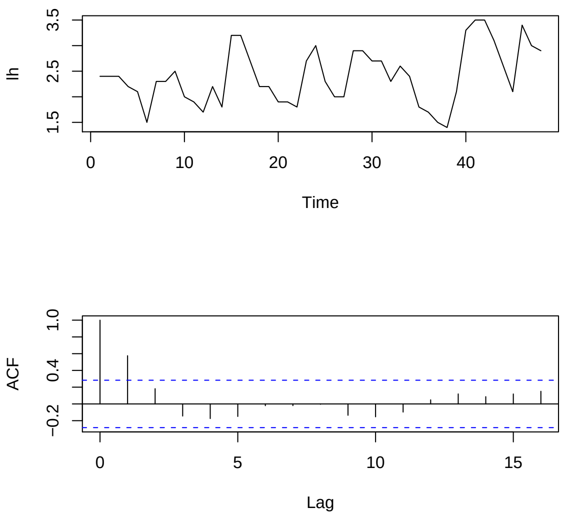

Autocorrelation

Sample Autocorrelation Coefficients

They measure the correlation between observations different distances apart.

$$r_k = \frac{\sum_{t=1}^{N-k}(x_t - \bar{x})(x_{t+k} - \bar{x})}{\sum_{t=1}^{N}(x_t - \bar{x})^2}$$

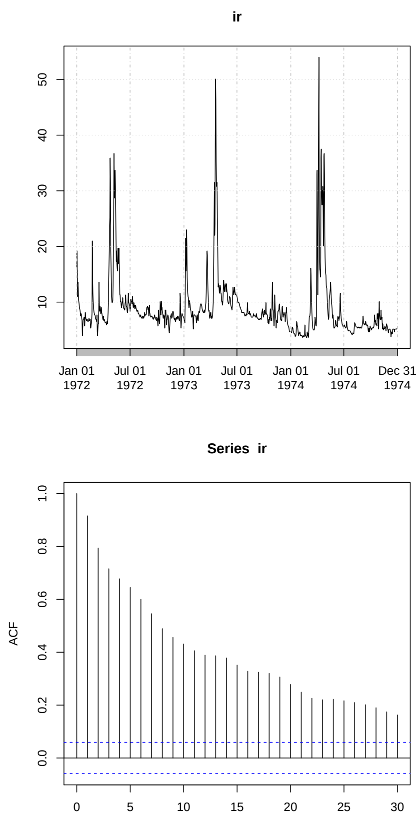
Correlogram



Plot the sample autocorrelation coefficients against the lags, $k = 0, 1, \cdots, M$.

Correlograms in R

```
> par(mfrow=c(2,1))
> plot(ir)
> acf(ir)
```



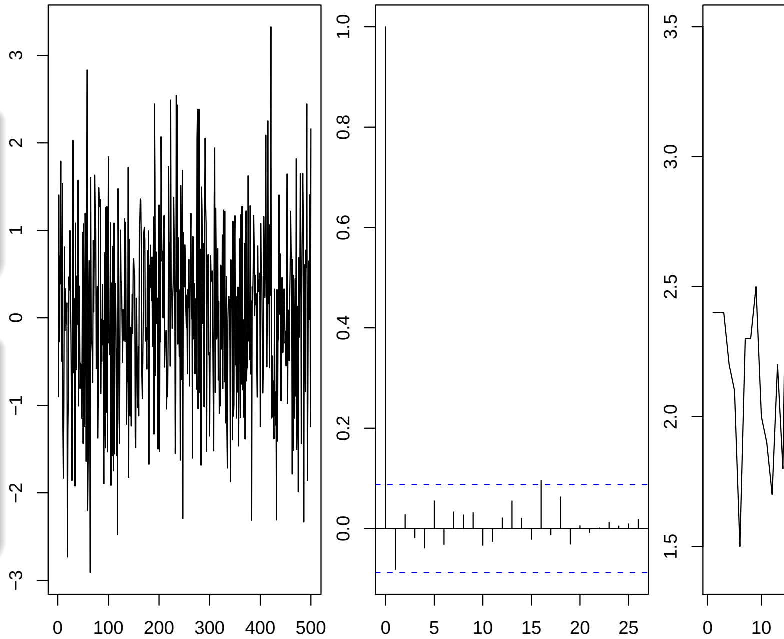
Interpreting the Correlogram

Random Series

Most r_k's near 0. Still, it is possible that 1 on 20 is significant...

Short-Term Correlation

Fairly large value of r_1 with sucessive values rapidly tending to non-significant.



Interpreting the Correlogram (cont.)

Alternating Series

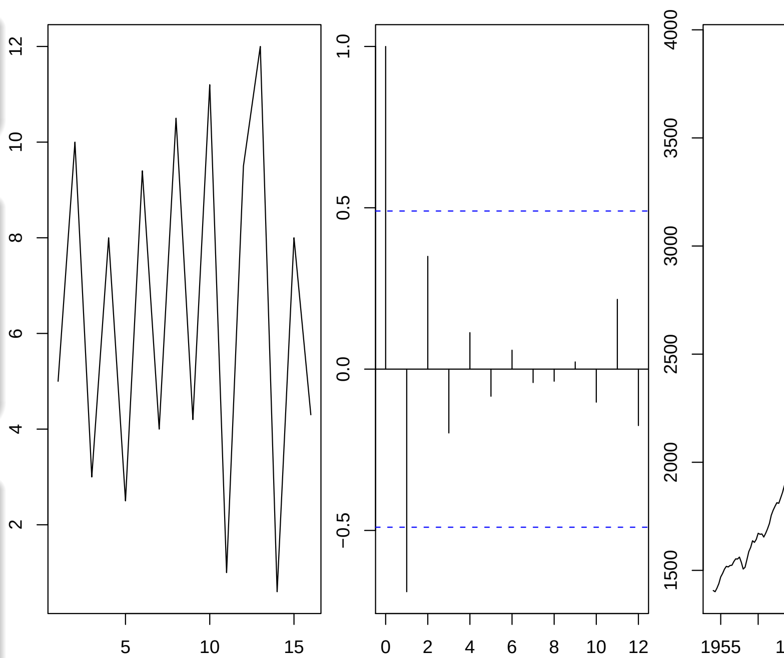
Similar pattern on the values of r_k.

Non-Stationary Series

For series with a trend the values of r_k will not go down till very large values of the lag.

Seasonal Series

The correlagram tends to exhibit the same periodicity as the original series.



Handling Real World Data

A Check List of Common Sense Things to Do (taken from Chatfield, 2004)

- Do you understand the context? Have the right variables been measured?
- Have all the time series been plotted?
- Are there missing values? If so, what should be done about them?
- Are there any outliers? If so, what should be done about them?
- Are there any discontinuities? If so, what do they mean?
- Does it make sense to transform the variables?
- Is trend present? If so, what should be done about it?
- Is seasonality present? If so, what should be done about it?

Chatfield, C. (2004): The Analysis of Time Series - an introduction. CRC.

Goals of an Evaluation Method

- The golden rule:

  The data used for evaluating (or comparing) any models cannot be seen during model development.

- The goal of any evaluation procedure:
  - Obtain a reliable estimate of some evaluation measure.
    High probability of achieving the same score on other samples of the same population.
- Evaluation Measures
  - Predictive accuracy.
  - Model size.
  - Computational complexity.

Performance Estimation for Time Series Models

- The usual techniques for model evaluation revolve around resampling.
  - Simulating the reality.
    - Obtain an evaluation estimate for unseen data.
- Examples of Resampling-based Methods
  - Holdout.
  - Cross-validation.
  - Bootstrap.

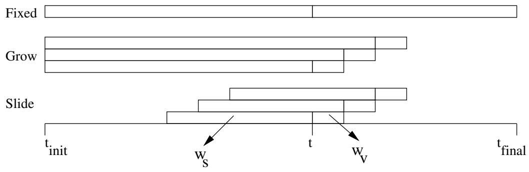
Time Series Data Are Special!

Any form of resampling changes the natural order of the data!

Correct Evalution of Time Series Models

- General Guidelines
  - Do not "forget" the time tags of the observations.
  - Do not evaluate a model on past data.
- A possible method
  - Divide the existing data in two time windows
    - Past data (observations till a time t).
    - "Future" data (observations after t).
  - Use one of these three learn-test alternatives
    - Fixed learning window.
    - Growing window.
    - Sliding window.

Learn-Test Strategies



Fixed Window

A single model is obtained with the available "training" data, and applied to all test period.

Growing Window

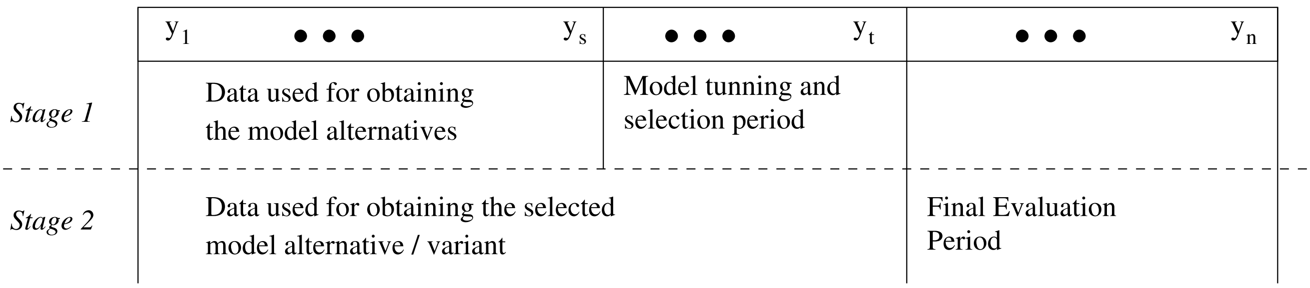
Every w_v test cases a new model is obtained using all data available till then.

Sliding Window

Every w_v test cases a new model is obtained using the previous w_s observations of the time series.

Dealing with model selection

- Most modelling techniques involve some form of parameters that usually need to be tunned.
- The following describes an evaluation methodology considering this issue:



Some Metrics for Evaluating Predictive Performance

Absolute Measures

- Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - x_i)^2$$

- Mean Absolute Deviation (MAD)

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|\hat{x}_i - x_i|$$

Relative Measures

- Theil Coefficient

$$U = \frac{\sqrt{\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}}{\sqrt{\sum_{i=1}^{n}(x_i - x_{i-1})^2}}$$

- Mean Absolute Percentage Error (MAPE)

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{(\hat{x}_i - x_i)}{x_i}\right|$$

The Goal of an Experimental Comparison

- Given a set of observations of a time series X.
- Given a set of alternative modelling approaches M.
- Obtain estimates of the predictive performance of each m_i for this time series.

More specifically,
given a forecasting period size, w_{test},
and a predictive performance statistic, Err,
we want to obtain a reliable estimate of the value of Err
for each m_i.

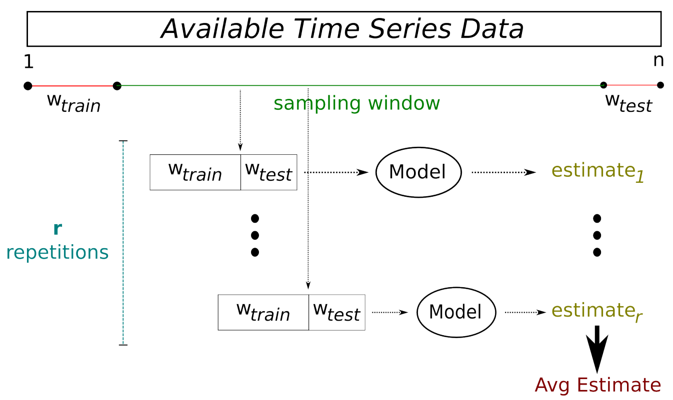
Using Monte Carlo Simulations for Obtaining Reliable Estimates of *Err*

- A possible approach would be to use our proposed method of Model Selection.
- This would give us one estimate of *Err*.
- More reliability is achievable if more repetitions of the process are carried out.

Using Monte Carlo Simulations for Obtaining Reliable Estimates of *Err*

Monte Carlo Estimates for Time Series Forecasting

Given: a time series, a training window size, w_{train}, a testing window size, w_{test}, and a number of repetitions, r,
- randomly generate r points in the interval $]w_{train}, (n - w_{test})[$,
- for each point learn a model with data in interval $[r - w_{train}, r]$ and test it with the data in the interval $[r + 1, r + w_{test}]$



Assumptions of “Classical” Linear Approaches

- *Linearity*
  The model of the time series behaviour is linear on its inputs.
- *Stationarity*
  The underlying equations governing the behaviour of the system do not change with time.

Most “classical” approaches assume stationary time series, thus one usually needs to transform non-stationary time series into stationary ones before using these tools.

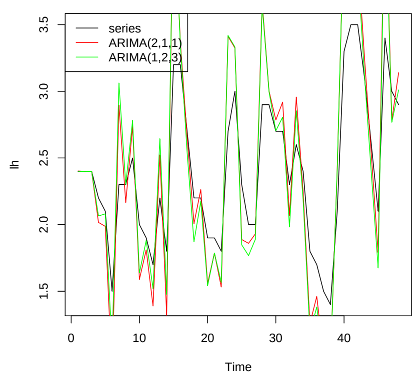
Integrated ARMA (or ARIMA) Models

Definition

An integrated ARMA (or ARIMA) model of order p, d, q is a series given by

$W_t = \sum_{i=0}^{p} \alpha_i W_{t-i} + \frac{1}{q} \sum_{i=0}^{q} X_{t-i}$

where $W_t = \nabla^d X_t$ is a d order difference.

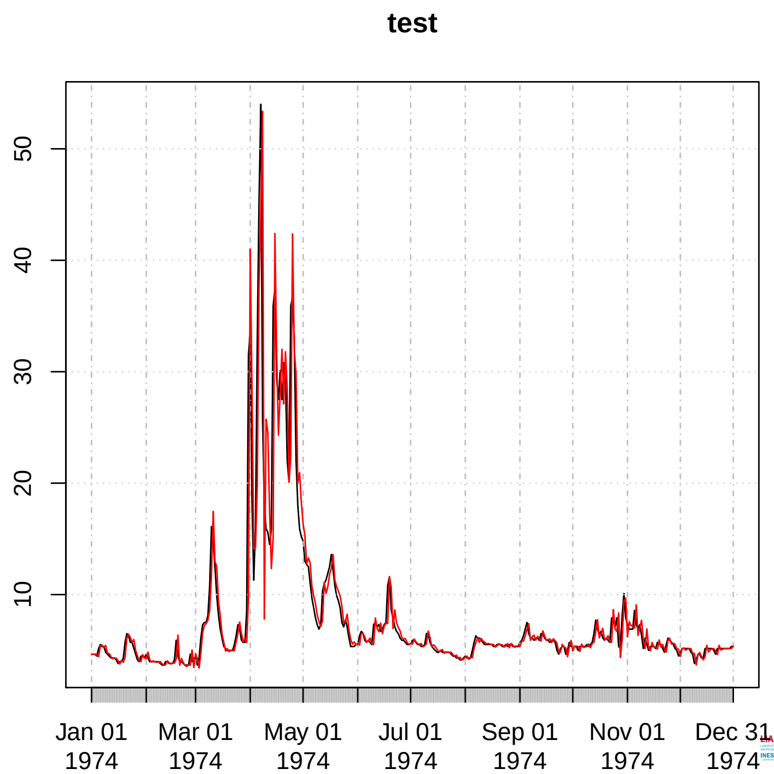


ARIMA models in R

```
> train <- ir['/1973-12-31']
> test <- ir['1974-01-01/']
>
> mad <- function(t,p) mean(abs(t-p))
>
>
> prevsARIMA <- function(tr, ordem) {
+   modelo <- arima0(as.vector(tr), ordem)
+   as.vector(predict(modelo, n.ahead = 1)$pred)
+ }
>
> ARIMA <- function(train, test, ord) {
+   pre <- rollapply(c(train, test), length(train), prevsARIMA,
+                    align = "right", ordem = ord)
+   as.xts(lag(pre, -1))
+ }
>
> prevs <- ARIMA(train,test,c(3,1,2))
> mad(test,prevs)
[1] 1.040762
```


ARIMA models in R - 2

```
> plot(test)
> lines(prevs,col='red')
```



Delay-Coordinate Embedding

Theorem (Takens, 1981)

Informally, it states we can uncover the dynamics of any time series given the information on e past values of the series. For that to be possible we need to know the correct embed size (how far back in time to look)

An Example of Delay-Coordinate Embedding

Example

Given the time series, $y_1, y_2, y_3, \cdots, y_{100}$, an embed dimension of 5, the resulting embed vectors are,

$$\begin{aligned} r_5 &= < y_5, y_4, y_3, y_2, y_1 > \\ r_6 &= < y_6, y_5, y_4, y_3, y_2 > \\ r_7 &= < y_7, y_6, y_5, y_4, y_3 > \\ r_8 &= < y_8, y_7, y_6, y_5, y_4 > \\ &\dots \end{aligned}$$

Consequences of Delay-Coordinate Embedding

If the system dynamics can be captured by a certain embed, then we may try to model the relationship between the state of the system and the future values of the series.

That is, we can try to obtain a model of the form,

$$Y_{t+h} = f(r_t)$$

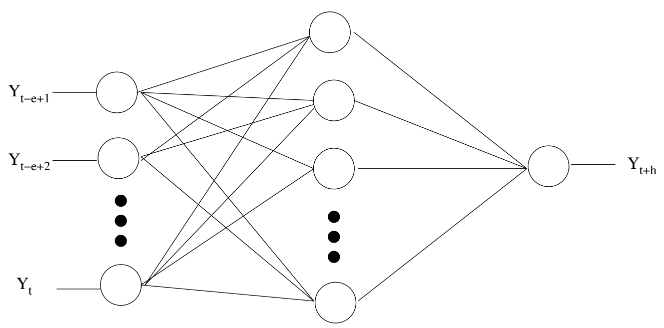
This modelling task can be handled by any multiple regression tool we have studied before!

Time Series Models | Non-Linear Approaches

An Example

Using Artificial Neural Networks (ANNs)

- ANNs are highly non-linear modelling techniques.
- Given the embedding representation we may use an ANN for our modelling task.
- Such network can be used to obtain predictions for Y_{t+h}, where h is the forecasting horizon, given the current embed.

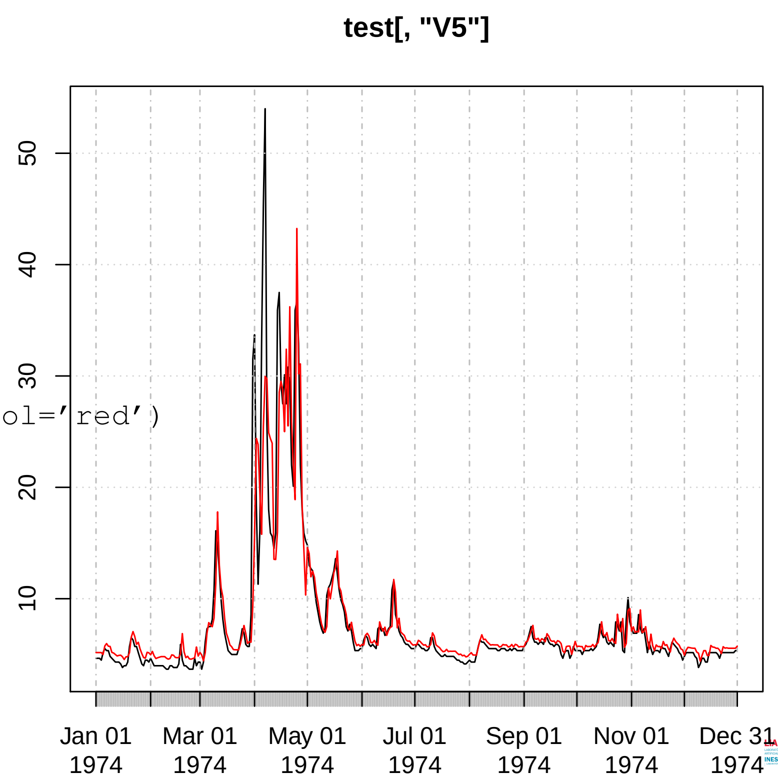


An Example with SVMs

```
> create.data <- function(ts,embed) {
+   t <- index(ts)[-(1:(embed-1))]
+   e <- embed(ts,embed)
+   colnames(e) <- paste('V',embed:1,sep='')
+   xts(e,t)
+ }
> ds <- create.data(ir,5)
> train <- ds['/1973-12-31']
> test <- ds['1974-01-01/']
>
> library(e1071)
>
> m <- svm(V5 ~ .,train,cost=10)
> p.s <- predict(m,test)
> mad(test,p.s)
[1] 1.409197
```

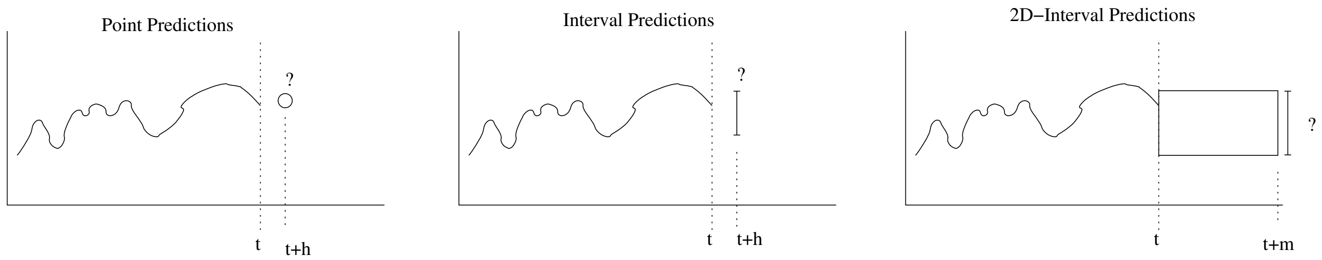

SVMs - 2

```
plot(test[,'V5'])
lines(xts(p.s,index(test)),col='red')
```



2D-Interval Predictions for Time Series
What?

Goal
Forecast the range of plausible values of a time series for a future time interval

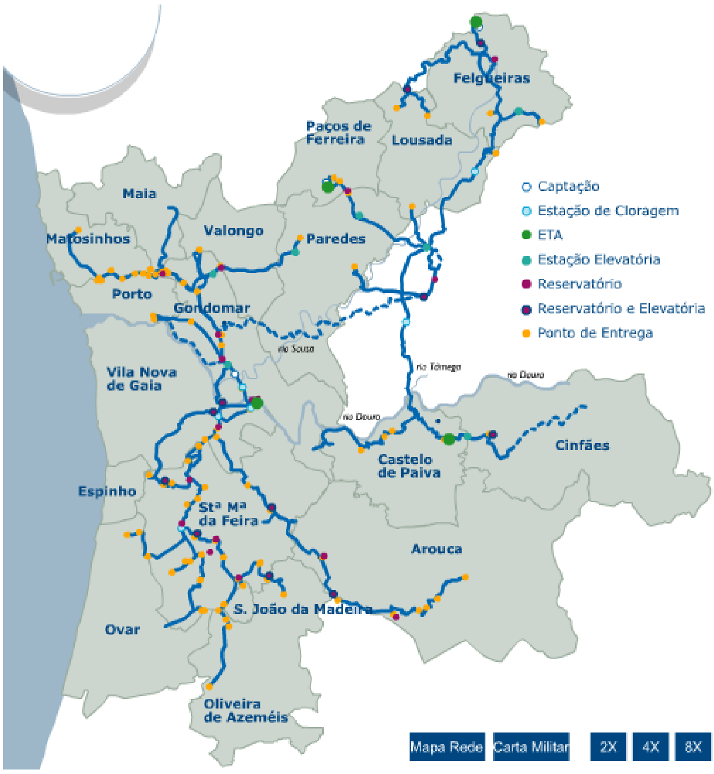


2D-Interval Predictions for Time Series
Why?

Motivation
- Several applications require planning ahead based on forecasts (e.g. production planning based on sales forecast)
- Other applications require decisions to be made based on predictions of expected scenarios for some future time interval (e.g. financial investments)
- Our work was driven by a particular application : water quality control on a large distribution network

Monitoring Water Quality Parameters

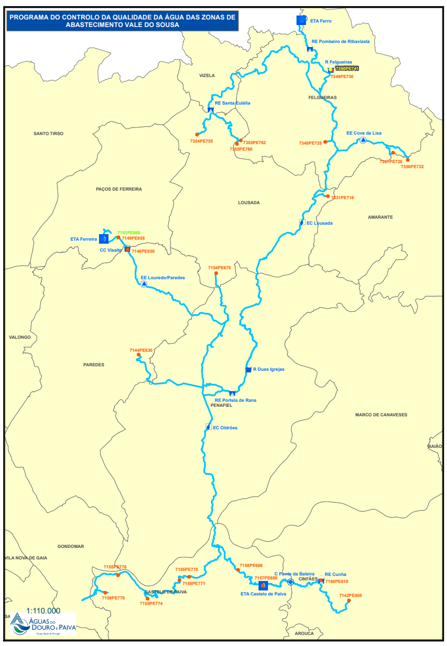
AdDP company



- Hundreds of parameters have to be monitored
- Legal limits with heavy fines
- In-company tighter limits for generating alarms and inspection decisions
- Very dynamic system - notion of "normality" varies along the year

Monitoring Water Quality Parameters

Application Goals



- At the beginning of each month provide an interval of expected values for a set of parameters (interval of "normality")
- Values outside these intervals should generate alarms and lead to inspection actions
- These alarms and inspections may lead to preventive actions to avoid surpassing the legal limits

Monitoring Water Quality Parameters

Defining the Data Mining Task

- What we want: provide an interval of expected values for a set of time series (interval of "normality")
- Can be seen as a form of summary statistic of the unknown future distribution of values of the series
- We will use the interquartile range as target summary statistic
  - Based on the 1st and 3rd quartiles
  - Roughly 50% of the cases are supposed to be inside that range
- Summarizing: our task will be to obtain estimates of the 1st and 3rd quartiles for a future time interval

Obtaining Predictions for a Future Time Interval

Possible Approaches with Existing Work

- Iterated Predictions
  - At time t obtain a prediction for time $t+1$
  - Use this prediction as if it was past and obtain a prediction for $t+2$
  - Iterate this process until we have predictions for the target interval $[t+1, t+k]$
  - With the k predictions calculate the 1st and 3rd quartiles to obtain the interval of values
  - *Potential Drawback*: Accumulate errors
- $K-$models
  - Obtain k different models, each "designed" to predict the value $t+i$, where $i \in [1, k]$
  - With the k predictions of the k models calculate the 1st and 3rd quartiles to obtain the interval of values
  - *Potential Drawback*: Computational complexity for large values of k or online scenarios

Our Proposal in a Nutshell

The Key Idea

Directly predict the summary statistics instead of the future values of the series

Motivation

Quantiles are robust statistics with a distribution that is smoother than the original series. Our hypothesis is that predicting them should be easier.

More Formally...

Formalization

Let Q_α^k and Q_β^k be the α and β unknown quantiles of the time series values for a future time window of size k.
Define the following prediction problems:
$Q_\alpha^k = f(v_1, \cdots, v_a)$ and
$Q_\beta^k = f(v_1, \cdots, v_a)$,
where $v_1, \cdots, v_a$ are a set of descriptor variables.

Experimental Setup

Goal

Compare our approach (`quantiles`) with the two other approaches (`iterated` and `k-models`)

Used Predictive Models

Random Walk (`RW`), Regression Trees (`RT`), SVMs (`SVM`), Random Forests (`RF`) and Quantile Random Forests (`QRF`)

Experimental Setup (2)

Estimation Method

Monte Carlo simulation with 10 repetitions at randomly selected points in time. Estimates for different values of the future time window (k)

Data

All alternatives using the same predictor variables and model settings. Only difference is on the way the predictions for the 1st and 3rd quartiles are obtained.

Experimental Setup (3)

Evaluation Metrics

- Total Quantile Error (*TQE*)
- Mean Absolute Quantile error (*MAQ*)
- Total Utility of predictions (*Utility*)

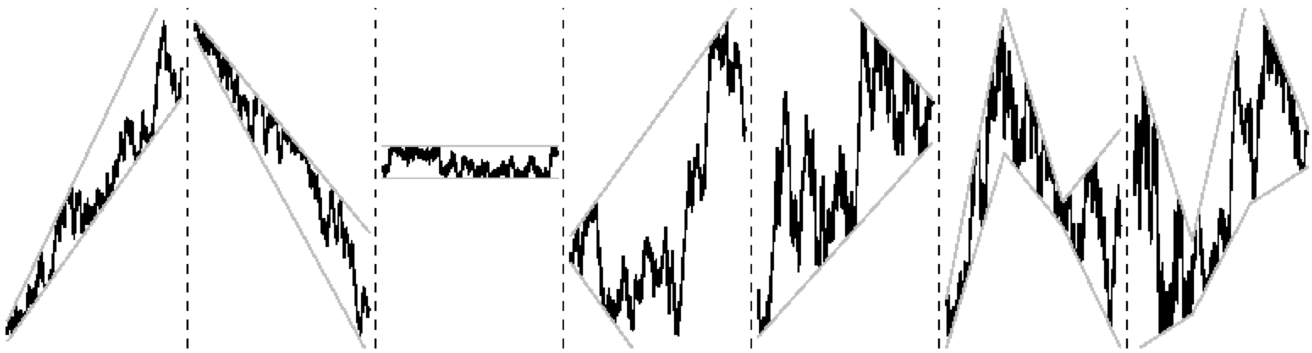
$$L_\alpha(y,\hat{y}) = \begin{cases} \alpha \cdot (y-\hat{y}) & \text{if} y \geq \hat{y} \\ (1-\alpha)\cdot(\hat{y}-y) & \text{otherwise} \end{cases}$$

$$TQE = \sum_{i=1}^{n} \left[\sum_{j=i}^{i+k} L_{0.25}(y_j, \hat{Q}^k_{0.25,i}) + \sum_{j=i}^{i+k} L_{0.75}(y_j, \hat{Q}^k_{0.75,i}) \right]$$

$$MAQ = \frac{1}{2n} \left[\sum_{i=1}^{n} |Q^k_{0.25,i} - \hat{Q}^k_{0.25,i}| + |Q^k_{0.75,i} - \hat{Q}^k_{0.75,i}| \right]$$

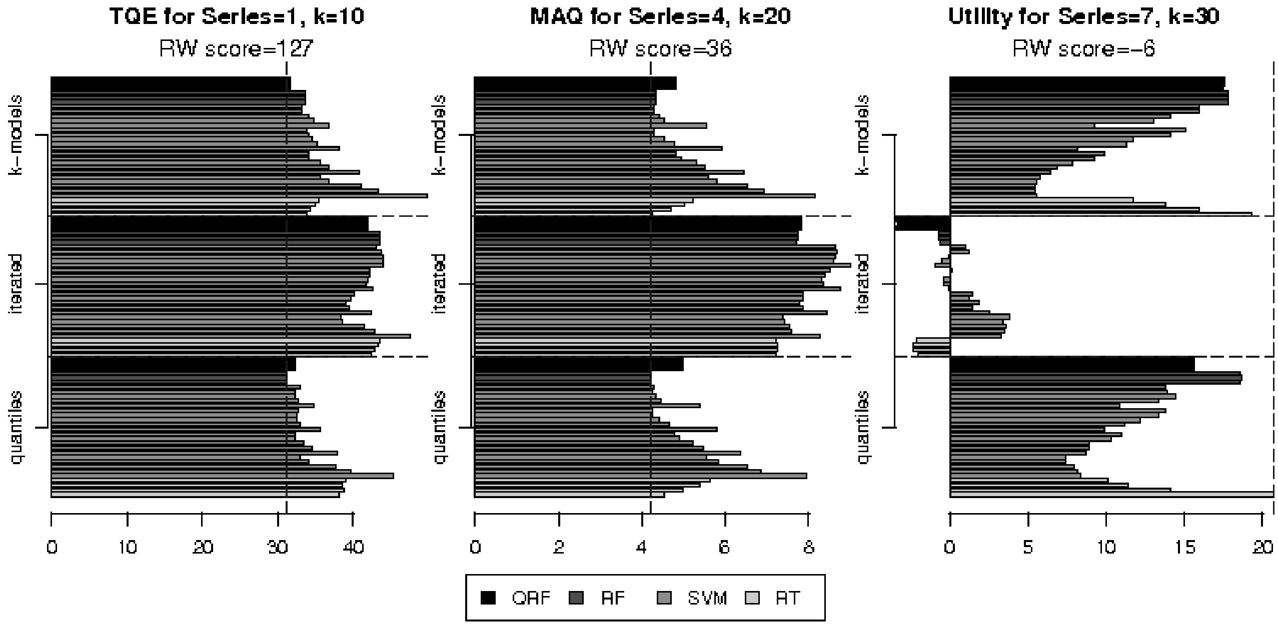
	low	*normal*	*high*

Experiments with Artificial Time Series

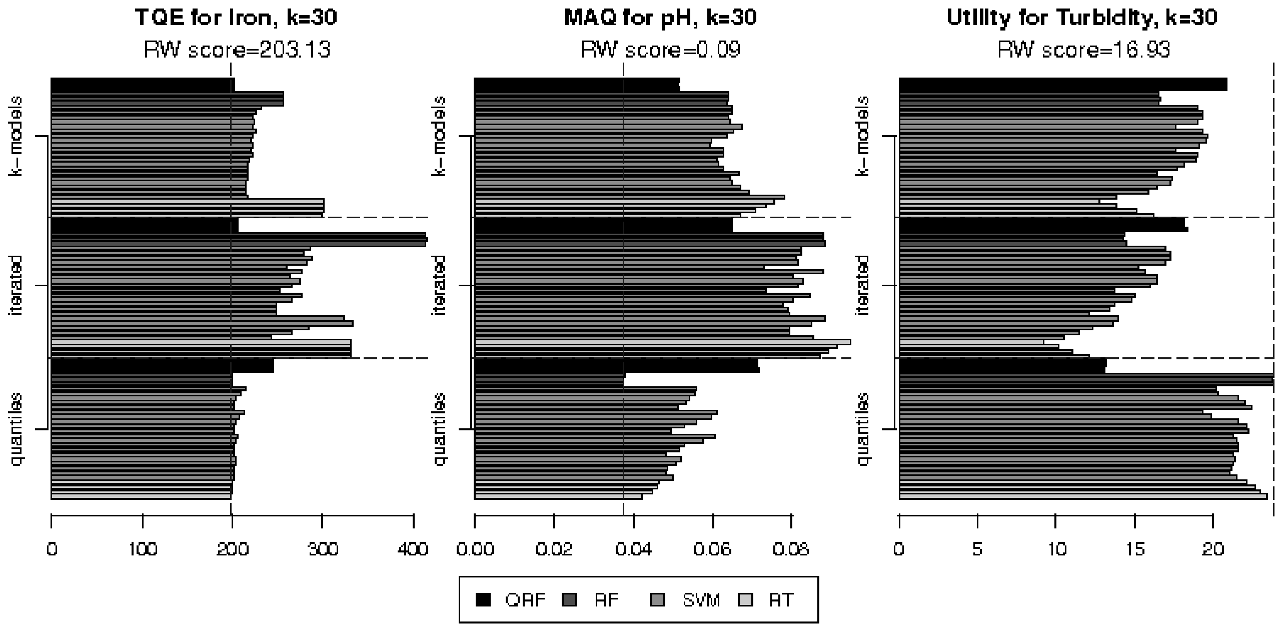


$$TGT = f(Y_t, \cdots, Y_{t-9}, Q^{-k}_{0.25,t}, Q^{-k}_{0.75,t}, \bar{Y}^{-k}, \sigma_Y^{-k})$$

Results with Artificial Time Series



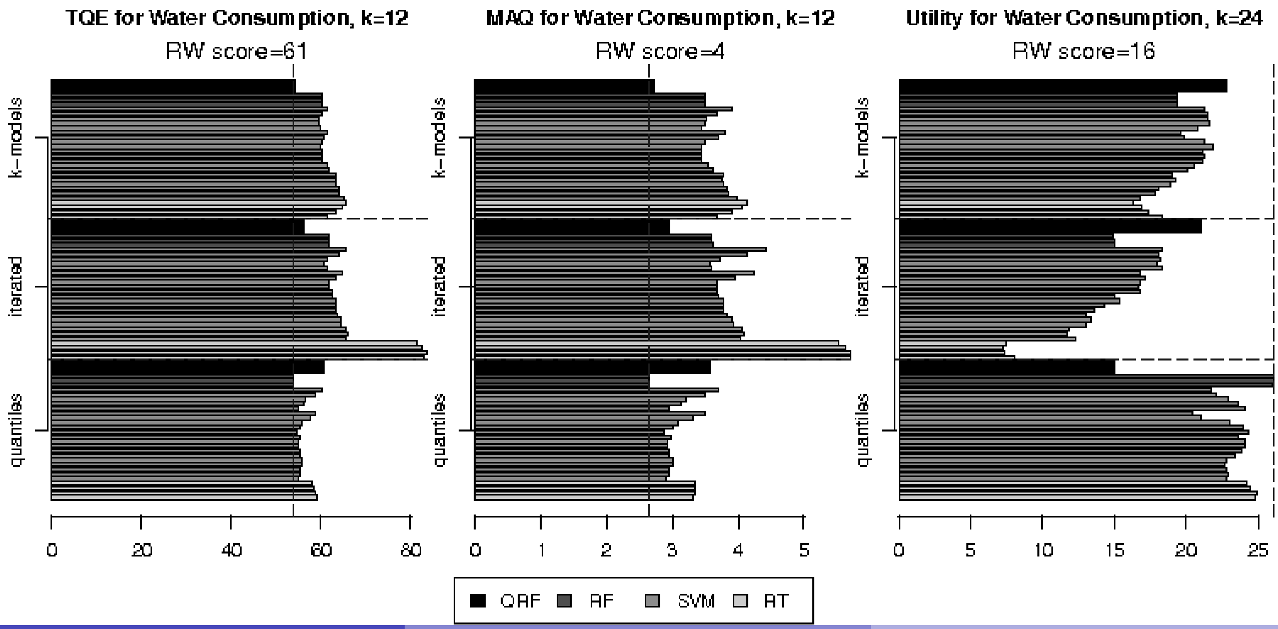
Results with Water Quality Parameters



Results with Water Demand Forecasting

Goal

Forecast the interval of plausible values for the water demand in the network for the next 12 and 24 hours.



2D-Interval Predictions for Time Series | Experimental Evaluation

Further Information

Full details: L. Torgo and O. Ohashi (2011) : 2D-Interval Predictions for Time Series, in Proceedings of 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'2011)

- All code and data of the KDD paper and also the full report of all experiments that were carried out are available at `http://www.dcc.fc.up.pt/~ltorgo/KDD`

Summary/Conclusions

- New type of forecasting tasks for time series with high applicability
- Proposed a method to address these tasks
- Encouraging results in rather different setups when taking both accuracy and computation time into account