R & Data Mining

Time Series Analysis and Mining with R

Yanchang Zhao

RDataMining.com
http://www.rdatamining.com/

18 July 2011

R & Data Mining

Outline

R

- a free software environment for statistical computing and graphics
- runs on Windows, Linux and MacOS
- widely used in academia and research, as well as industrial applications
- over 3,000 packages
- CRAN Task View: Time Series Analysis
  `http://cran.r-project.org/web/views/TimeSeries.html`

Time Series Data in R

- class `ts`
- represents data which has been sampled at equispaced points in time
- frequency=7: a weekly series
- frequency=12: a monthly series
- frequency=4: a quarterly series

Time Series Data in R


```
> a <- ts(1:20, frequency=12, start=c(2011,3))
> print(a)
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov
2011           1   2   3   4   5   6   7   8   9
2012  11  12  13  14  15  16  17  18  19  20
     Dec
2011  10
2012
> str(a)
 Time-Series [1:20] from 2011 to 2013: 1 2 3 4 5 6 7 
> attributes(a)
$tsp
[1] 2011.167 2012.750   12.000

$class
```

Outline

What is Time Series Decomposition

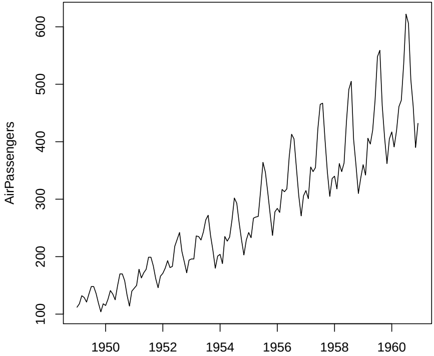
To decompose a time series into components:

- Trend component: long term trend
- Seasonal component: seasonal variation
- Cyclical component: repeated but non-periodic fluctuations
- Irregular component: the residuals

Data AirPassengers

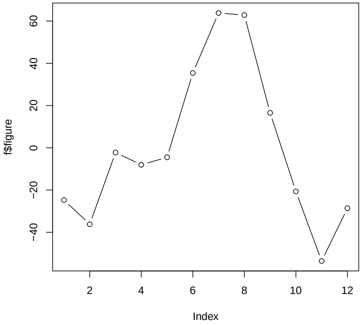
Data `AirPassengers`: Monthly totals of Box Jenkins international airline passengers, 1949 to 1960. It has 144(=12×12) values.

```
> plot(AirPassengers)
```



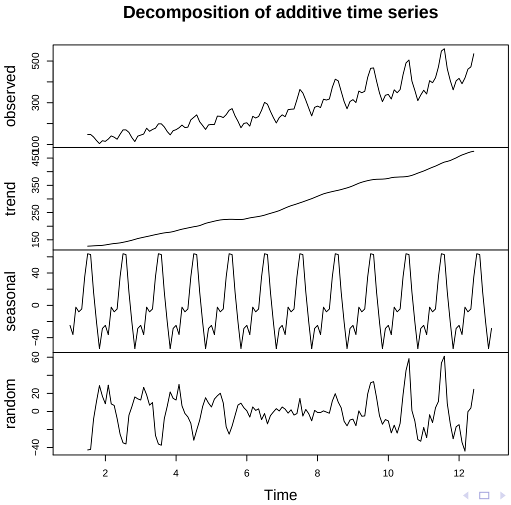
Decomposition

```
> apts <- ts(AirPassengers, frequency = 12)
> f <- decompose(apts)
> # seasonal figures
> plot(f$figure,type="b")
```



Decomposition


```
> plot(f)
```



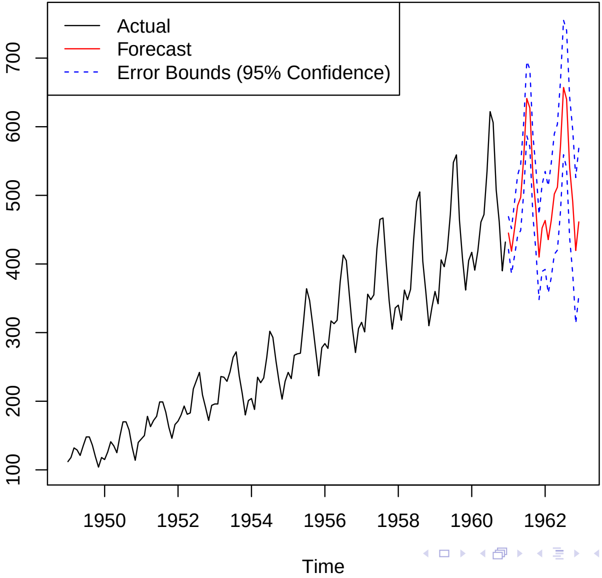
Outline

Time Series Forecasting

- To forecast future events based on known past data
- E.g., to predict the opening price of a stock based on its past performance
- Popular models
  - Autoregressive moving average (ARMA)
  - Autoregressive integrated moving average (ARIMA)

```
> # build an ARIMA model
> fit <- arima(AirPassengers, order=c(1,0,0),
+         list(order=c(2,1,0), period=12))
> fore <- predict(fit, n.ahead=24)
> # error bounds at 95% confidence level
> U <- fore$pred + 2*fore$se
> L <- fore$pred - 2*fore$se
> ts.plot(AirPassengers, fore$pred, U, L,
+          col=c(1,2,4,4), lty = c(1,1,2,2))
> legend("topleft", col=c(1,2,4), lty=c(1,1,2),
+         c("Actual", "Forecast",
+           "Error Bounds (95% Confidence)"))
```

Forecasting



Outline

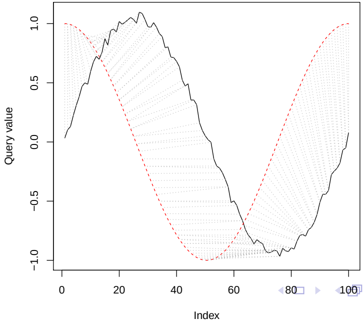
Time Series Clustering

- To partition time series data into groups based on *similarity* or *distance*, so that time series in the same cluster are similar
- Measure of distance/dissimilarity
    - Euclidean distance
    - Manhattan distance
    - Maximum norm
    - Hamming distance
    - The angle between two vectors (inner product)
    - Dynamic Time Warping (DTW) distance
    - ...

Dynamic Time Warping (DTW)

DTW finds optimal alignment between two time series.

```
> library(dtw)
> idx <- seq(0, 2*pi, len=100)
> a <- sin(idx) + runif(100)/10
> b <- cos(idx)
> align <- dtw(a, b, step=asymmetricP1, keep=T)
> dtwPlotTwoWay(align)
```



Synthetic Control Chart Time Series

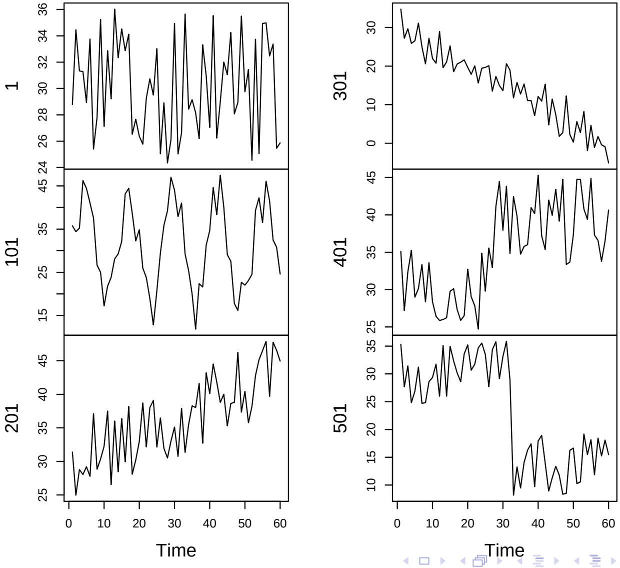
- The dataset contains 600 examples of control charts synthetically generated by the process in Alcock and Manolopoulos (1999).
- Each control chart is a time series with 60 values.
- Six classes:
  - 1-100 Normal
  - 101-200 Cyclic
  - 201-300 Increasing trend
  - 301-400 Decreasing trend
  - 401-500 Upward shift
  - 501-600 Downward shift
- http://kdd.ics.uci.edu/databases/synthetic_control/synthetic_control.html

Synthetic Control Chart Time Series


```
> # read data into R
> # sep="": the separator is white space, i.e., one
> # or more spaces, tabs, newlines or carriage return
> sc <- read.table("synthetic_control.data",
+                  header=F, sep="")
> # show one sample from each class
> idx <- c(1,101,201,301,401,501)
> sample1 <- t(sc[idx,])
> plot.ts(sample1, main="")
```

Six Classes

R and Time Series Data
Time Series Decomposition
Time Series Forecasting
Time Series Clustering
Time Series Classification
R Functions & Packages for Time Series
Conclusions

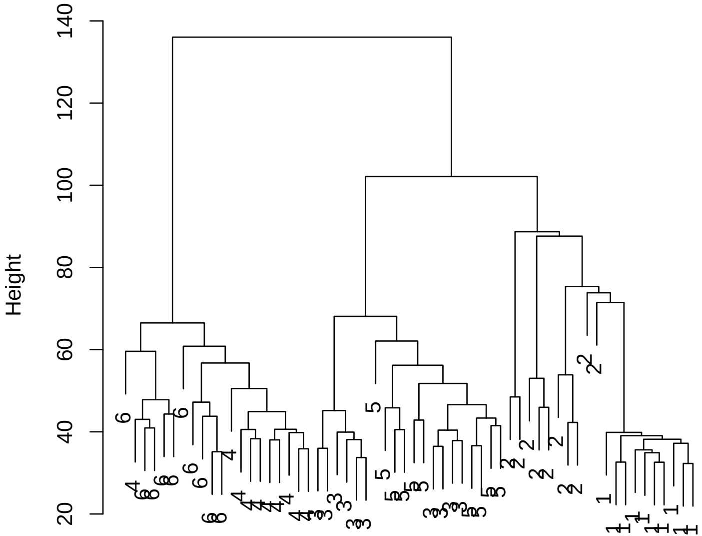


Hierarchical Clustering with Euclidean distance


```
> # sample n cases from every class
> n <- 10
> s <- sample(1:100, n)
> idx <- c(s, 100+s, 200+s, 300+s, 400+s, 500+s)
> sample2 <- sc[idx,]
> observedLabels <- c(rep(1,n), rep(2,n), rep(3,n),
+                     rep(4,n), rep(5,n), rep(6,n))
> # hierarchical clustering with Euclidean distance
> hc <- hclust(dist(sample2), method="ave")
> plot(hc, labels=observedLabels, main="")
```

Hierarchical Clustering with Euclidean distance

R and Time Series Data
Time Series Decomposition
Time Series Forecasting
Time Series Clustering
Time Series Classification
R Functions & Packages for Time Series
Conclusions



R & Data Mining

Hierarchical Clustering with Euclidean distance


```
> # cut tree to get 8 clusters
> memb <- cutree(hc, k=8)
> table(observedLabels, memb)
```

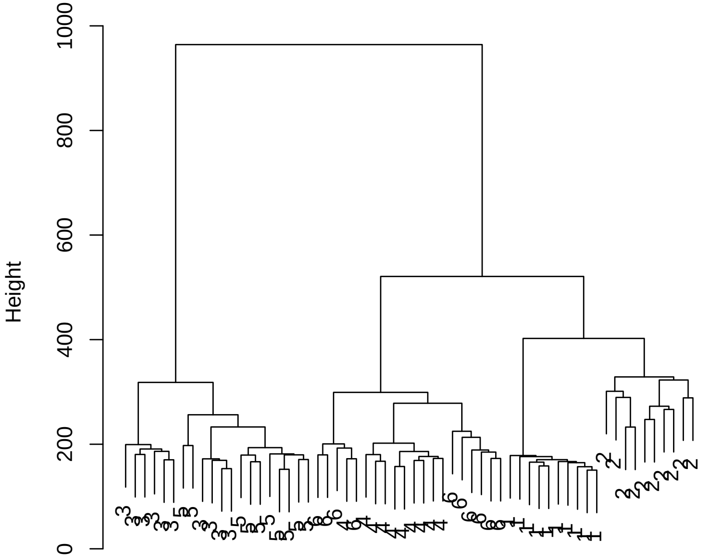
```
              memb
observedLabels  1  2  3  4  5  6  7  8
             1 10  0  0  0  0  0  0  0
             2  0  3  1  1  3  2  0  0
             3  0  0  0  0  0  0 10  0
             4  0  0  0  0  0  0  0 10
             5  0  0  0  0  0  0 10  0
             6  0  0  0  0  0  0  0 10
```

Hierarchical Clustering with DTW Distance


```
> myDist <- dist(sample2, method="DTW")
> hc <- hclust(myDist, method="average")
> plot(hc, labels=observedLabels, main="")
> # cut tree to get 8 clusters
> memb <- cutree(hc, k=8)
> table(observedLabels, memb)
```

```
              memb
observedLabels  1  2  3  4  5  6  7  8
             1 10  0  0  0  0  0  0  0
             2  0  4  3  2  1  0  0  0
             3  0  0  0  0  0  6  4  0
             4  0  0  0  0  0  0  0 10
             5  0  0  0  0  0  0 10  0
             6  0  0  0  0  0  0  0 10
```

Hierarchical Clustering with DTW Distance



Outline

Time Series Classification

Time Series Classification

- To build a classification model based on labelled time series
- and then use the model to predict the lable of unlabelled time series

Feature Extraction

- Singular Value Decomposition (SVD)
- Discrete Fourier Transform (DFT)
- Discrete Wavelet Transform (DWT)
- Piecewise Aggregate Approximation (PAA)
- Perpetually Important Points (PIP)
- Piecewise Linear Representation
- Symbolic Representation

Decision Tree (ctree)

ctree from package **party**

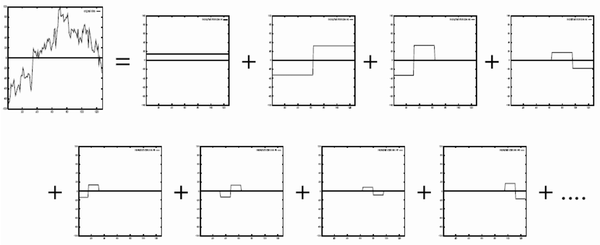
```
> classId <- c(rep("1",100), rep("2",100),
+              rep("3",100), rep("4",100),
+              rep("5",100), rep("6",100))
> newSc <- data.frame(cbind(classId, sc))
> library(party)
> ct <- ctree(classId ~ ., data=newSc,
+             controls = ctree_control(minsplit=20,
+                        minbucket=5, maxdepth=5))
```

Decision Tree

```
> pClassId <- predict(ct)
> table(classId, pClassId)

         pClassId
classId   1   2   3   4   5   6
      1 100   0   0   0   0   0
      2   1  97   2   0   0   0
      3   0   0  99   0   1   0
      4   0   0   0 100   0   0
      5   4   0   8   0  88   0
      6   0   3   0  90   0   7

> # accuracy
> (sum(classId==pClassId)) / nrow(sc)

[1] 0.8183333
```

DWT (Discrete Wavelet Transform)

- Wavelet transform provides a multi-resolution representation using wavelets.
- Haar Wavelet Transform – the simplest DWT
  http://dmr.ath.cx/gfx/haar/



- DFT (Discrete Fourier Transform): another popular feature extraction technique

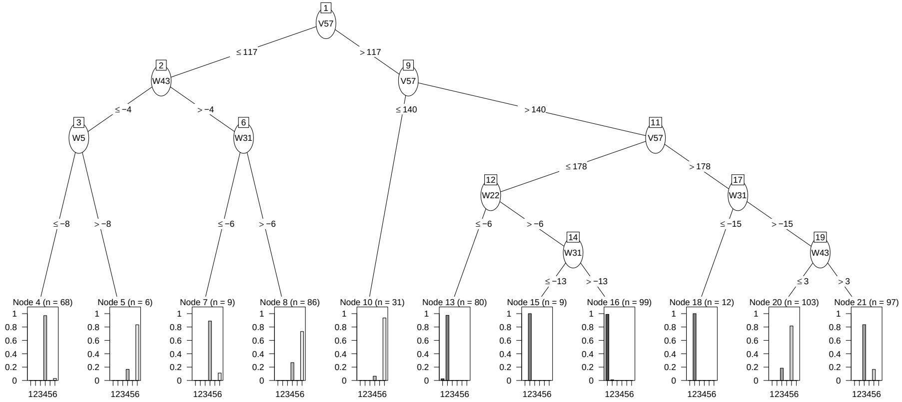
DWT (Discrete Wavelet Transform)


```
> # extract DWT (with Haar filter) coefficients
> library(wavelets)
> wtData <- NULL
> for (i in 1:nrow(sc)) {
+   a <- t(sc[i,])
+   wt <- dwt(a, filter="haar", boundary="periodic")
+   wtData <- rbind(wtData,
+       unlist(c(wt@W, wt@V[[wt@level]])))
+ }
> wtData <- as.data.frame(wtData)
> wtSc <- data.frame(cbind(classId, wtData))
```

Decision Tree with DWT


```
> ct <- ctree(classId ~ ., data=wtSc, controls =
+               ctree_control(minsplit=20,
+               minbucket=5, maxdepth=5))
> pClassId <- predict(ct)
> table(classId, pClassId)
       pClassId
classId  1  2  3  4  5  6
      1 98  2  0  0  0  0
      2  1 99  0  0  0  0
      3  0  0 81  0 19  0
      4  0  0  0 74  0 26
      5  0  0 16  0 84  0
      6  0  0  0  3  0 97
> (sum(classId==pClassId)) / nrow(wtSc)
[1] 0.8883333
```

```
> plot(ct, ip_args=list(pval=FALSE), ep_args=list(digits=0))
```



k-NN Classification

- find the k nearest neighbours of a new instance
- label it by majority voting
- needs an efficient indexing structure for large datasets

```
> k <- 20
> newTS <- sc[501,] + runif(100)*15
> distances <- dist(newTS, sc, method="DTW")
> s <- sort(as.vector(distances), index.return=TRUE)
> # class IDs of k nearest neighbours
> table(classId[s$ix[1:k]])
```

```
 4  6
 3 17
```

Results of Majority Voting

Label of newTS ← class 6

Outline

Functions - Construction, Plot & Smoothing

Construction

- `ts()` create time-series objects (*stats*)

Plot

- `plot.ts()` plot time-series objects (*stats*)

Smoothing & Filtering

- `smoothts()` time series smoothing (*ast*)
- `sfilter()` remove seasonal fluctuation using moving average (*ast*)

Functions - Decomposition & Forecasting

Decomposition

- `decomp()` time series decomposition by square-root filter (*timsac*)
- `decompose()` classical seasonal decomposition by moving averages (*stats*)
- `stl()` seasonal decomposition of time series by `loess` (*stats*)
- `tsr()` time series decomposition (*ast*)
- `ardec()` time series autoregressive decomposition (*ArDec*)

Forecasting

- `arima()` fit an ARIMA model to a univariate time series (*stats*)
- `predict.Arima()` forecast from models fitted by `arima` (*stats*)

Packages

- **timsac** time series analysis and control program
- **ast** time series analysis
- **ArDec** time series autoregressive-based decomposition
- **ares** a toolbox for time series analyses using generalized additive models
- **dse** tools for multivariate, linear, time-invariant, time series models
- **forecast** displaying and analysing univariate time series forecasts
- **dtw** Dynamic Time Warping – find optimal alignment between two time series
- **wavelets** wavelet filters, wavelet transforms and multiresolution analyses

Online Resources

- An R Time Series Tutorial

  http://www.stat.pitt.edu/stoffer/tsa2/R_time_series_quick_fix.htm
- Time Series Analysis with R

  http://www.statoek.wiso.uni-goettingen.de/veranstaltungen/zeitreihen/sommer03/ts_r_intro.pdf
- Using R (with applications in Time Series Analysis)

  http://people.bath.ac.uk/masgs/time%20series/TimeSeriesR2004.pdf
- CRAN Task View: Time Series Analysis

  http://cran.r-project.org/web/views/TimeSeries.html
- R Functions for Time Series Analysis

  http://cran.r-project.org/doc/contrib/Ricci-refcard-ts.pdf
- R Reference Card for Data Mining;
  R and Data Mining: Examples and Case Studies

  http://www.rdatamining.com/
- Time Series Analysis for Business Forecasting

  http://home.ubalt.edu/ntsbarsh/stat-data/Forecast.htm

Outline

R & Data Mining

Conclusions

- Time series decomposition and forecasting: many R functions and packages available
- Time series classification and clustering: no R functions or packages specially for this purpose; have to work it out by yourself
- Time series classification: extract and build features, and then apply existing classification techniques, such as SVM, k-NN, neural networks, regression and decision trees
- Time series clustering: work out your own distance/similarity metrics, and then use existing clustering techniques, such as k-means and hierarchical clustering
- Techniques specially for classifying/clustering time series data: a lot of research publications, but no R implementations (as far as I know)

The End

Email:	yanchangzhao@gmail.com
RDataMining:	`http://www.rdatamining.com`
Twitter:	`http://twitter.com/rdatamining`
Group on Linkedin:	`http://group.rdatamining.com`
Group on Google:	`http://group2.rdatamining.com`