Time series analysis with Quantile regression

Anton Antonov

10/1/2016

Introduction

This document (R-Markdown file) is made for the R-part of the MathematicaVsR project "Time series analysis with Quantile Regression".

The main goal of this document is to demonstrate how to do in R:

- getting weather data (or other time series data),
- fitting Quantile Regression (QR) curves to time series data, and
- using QR to find outliers and conditional distributions.

Libraries

```
library(weatherData)
library(ggplot2)
library(reshape2)
library(quantreg)
library(splines)
```

Getting time series data

Assume we want to obtain temperature time series data for Atlanta, Georgia, USA for the time interval from 2011.04.01 to 2016.03.31 .

Following the guide [2] we can download that weather data in the following way.

First we find weather stations identifiers in Atlanta, GA:

```
getStationCode("Atlanta")
```

```
## [[1]]
##            Station State airportCode
## 426        Atlanta    GA        KATL
## 432 Atlanta Fulton    GA        KFTY
## 440 Atlanta Dekalb    GA        KPDK
##
## [[2]]
## [1] "USA GA ATLANTA          KATL  ATL   72219  33 38N  084 27W  296   X     U     A    0 US"
## [2] "USA GA ATLANTA/FULTON   KFTY  FTY         33 47N  084 31W  263   X     T     A    3 US"
## [3] "USA GA ATLANTA/PAULDING KPUJ  PUJ         33 55N  084 56W  393   X           W    8 US"
## [4] "USA GA ATLANTA/RFC      KATR  ATR         33 22N  084 34W  312                   R 8 US"
## [5] "USA GA ATLANTA/ARTCC    KZTL  ZTL         33 23N  084 20W  312         A           8 US"
## [6] "USA GA ATLANTA RFC      KALR  ALR         33 22N  084 34W  248                   R 8 US"
```

Let use the first one "KATL". The following code downloads the temperature data for desired time interval.

```r
if(!exists("tempDF")) {
  res <-
    llply( seq(2011,2015), function(y) {
      getWeatherForDate( station_id = "KATL",
                         start_date = paste(y, "04-01", sep="-" ),
                         end_date = paste(y+1, "03-31", sep="-" ) )
    }, .progress = "None")
  tempDF <- do.call(rbind, res)
}
```

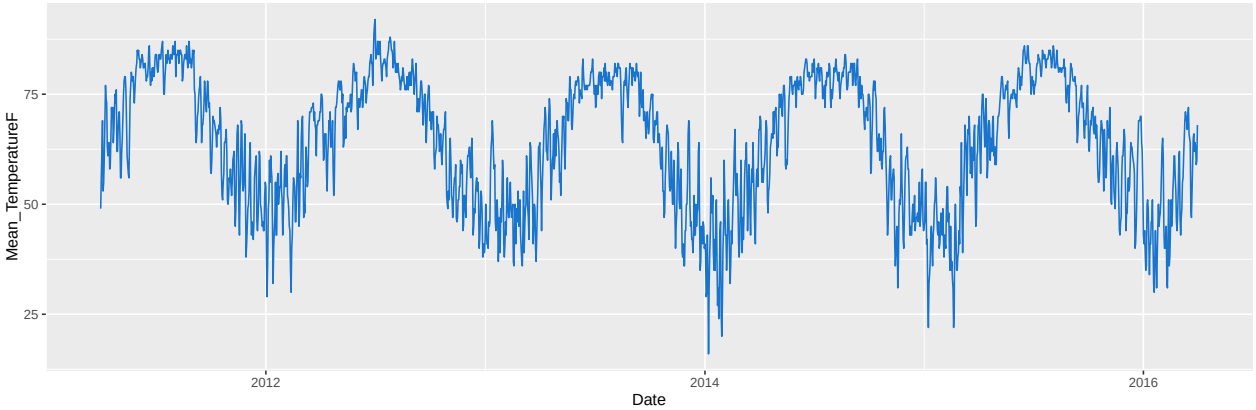
The obtained data frame has the following form:

```r
head(tempDF)
```

```
##         Date Max_TemperatureF Mean_TemperatureF Min_TemperatureF Index
## 1 2011-04-01               58                49               40     1
## 2 2011-04-02               69                56               43     2
## 3 2011-04-03               79                62               44     3
## 4 2011-04-04               83                69               55     4
## 5 2011-04-05               62                53               43     5
## 6 2011-04-06               71                55               39     6
##      AbsTime Index    AbsTime Index    AbsTime Index    AbsTime Index
## 1 1301630400     1 1301630400     1 1301630400     1 1301630400     1
## 2 1301716800     2 1301716800     2 1301716800     2 1301716800     2
## 3 1301803200     3 1301803200     3 1301803200     3 1301803200     3
## 4 1301889600     4 1301889600     4 1301889600     4 1301889600     4
## 5 1301976000     5 1301976000     5 1301976000     5 1301976000     5
## 6 1302062400     6 1302062400     6 1302062400     6 1302062400     6
##      AbsTime Index    AbsTime Index    AbsTime Index    AbsTime Index
## 1 1301630400     1 1301630400     1 1301630400     1 1301630400     1
## 2 1301716800     2 1301716800     2 1301716800     2 1301716800     2
## 3 1301803200     3 1301803200     3 1301803200     3 1301803200     3
## 4 1301889600     4 1301889600     4 1301889600     4 1301889600     4
## 5 1301976000     5 1301976000     5 1301976000     5 1301976000     5
## 6 1302062400     6 1302062400     6 1302062400     6 1302062400     6
##      AbsTime Index    AbsTime Index    AbsTime Index    AbsTime Index
## 1 1301630400     1 1301630400     1 1301630400     1 1301630400     1
## 2 1301716800     2 1301716800     2 1301716800     2 1301716800     2
## 3 1301803200     3 1301803200     3 1301803200     3 1301803200     3
## 4 1301889600     4 1301889600     4 1301889600     4 1301889600     4
## 5 1301976000     5 1301976000     5 1301976000     5 1301976000     5
## 6 1302062400     6 1302062400     6 1302062400     6 1302062400     6
##      AbsTime Index    AbsTime
## 1 1301630400     1 1301630400
## 2 1301716800     2 1301716800
## 3 1301803200     3 1301803200
## 4 1301889600     4 1301889600
## 5 1301976000     5 1301976000
## 6 1302062400     6 1302062400
```

Below we are going to use the mean temperatures. Here is plot of that time series data:

```r
ggplot(tempDF) +
  geom_line(aes(x = Date, y = Mean_TemperatureF), color='dodgerblue3')
```



(The color name was selected from the web page "ggplot2 Quick Reference: colour (and fill)".)

Fitting Quantile regression curves and finding outliers

QR fitting of B-splines

The package `quantreg` provides several ways (functions and work flow) to apply QR to time series data. In this document we interested in applying QR using B-spline basis functions. Following the vignette [1] this can be done in the following way.

First we are going to add to the time series data frame an index column and an absolute time column.

```
tempDF <- tempDF[order(tempDF$Date),]
tempDF <- cbind( tempDF, Index=1:nrow(tempDF), AbsTime = as.numeric(tempDF$Date) )
```

Next we make a model matrix for a selected number of knots.

```
nKnots <- 30
X <- model.matrix( Mean_TemperatureF ~ bs(Index, df = nKnots + 3, degree = 3), data = tempDF )
```

We find the QR curves – called regression quantiles – at these quantiles:

```
qs <- c(0.02,0.1,0.25,0.5,0.75,0.9,0.98)
```

Do the QR fit:

```
qcurves <-
  llply( qs, function(x) {
    fit <- rq( Mean_TemperatureF ~ bs(Index, df = nKnots + 3, degree = 3), tau = x, data = tempDF)
    X %*% fit$coef
  }, .progress = "none")
```

We put the QR fitting result into a data frame with which further manipulations and plotting would be easier.

```
qfitDF <- do.call(cbind, qcurves )
qfitDF <- data.frame(Index=1:nrow(qfitDF), Date = tempDF$Date, qfitDF )
```

Finding outliers

At this point finding the outliers is simple – we just pick the points (dates) with temperatures higher than the 0.98 regression quantile (multiplied by some factor close to 1, like, 1.005.)

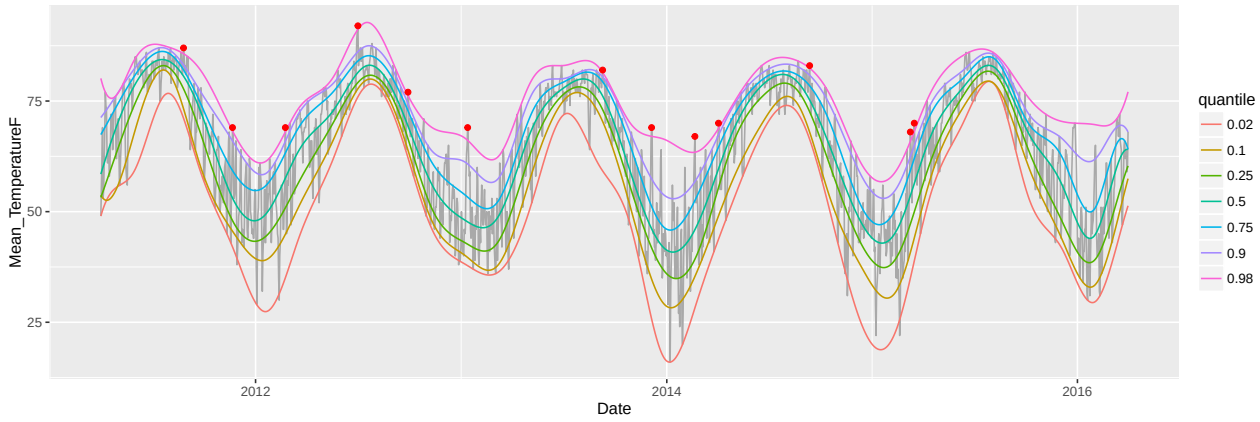
```
outlierInds <- which( tempDF$Mean_TemperatureF > 1.005 * qfitDF[,ncol(qfitDF)] )
```

Plot

The best way to plot the data is through melting into long form data frame. The identified outliers are given with red points.

```
names(qfitDF) <- c( "Index", "Date", qs )
qfitMeltedDF <- melt( data = qfitDF, id.vars = .(Date, Index) )
names(qfitMeltedDF) <- gsub( "variable", "quantile", names(qfitMeltedDF) )

ggplot( tempDF ) +
  geom_line( aes( x = Date, y = Mean_TemperatureF ), color = 'darkgrey' ) +
  geom_line( data = qfitMeltedDF, aes( x = Date, y = value, color = quantile ) ) +
  geom_point( data = tempDF[outlierInds, ], aes( x = Date, y = Mean_TemperatureF ), color = 'red')
```



Re-construction of conditional probabilities distributions

CDF and PDF re-construction function definitions

```
CDFEstimateFunction <- function( qs, qvals ) {
  ## splinefun( x = qvals, y = qs, method = "natural" )
  approxfun( x = qvals, y = qs, method = "linear" )
}
```

Since we deal with piece-wise linear functions for CDF the PDF has to defined ad-hoc instead of using functions that find derivatives.

```
PDFEstimateFunction <- function( qs, qvals ) {
  names(qvals) <- NULL; names(qs) <- NULL
  xs = ( qvals[-length(qvals)] + qvals[-1] ) / 2
  ys = diff(qs) / diff(qvals)
  approxfun( x = xs, y = ys, method = "constant" )
}
```

Note, that if we used `splinefun` for the calculation of the CDF function `cdfFunc` we could implement the PDF function simply as `pdfFunc <- function(x) cdfFunc( x, 1 )`.

QR with for lots of quantiles

Consider the quantiles:

```
qs <- seq(0,1,0.05); qs <- c(0.02, qs[qs > 0 & qs < 1 ], 0.98); qs
```

```
##  [1] 0.02 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65
## [15] 0.70 0.75 0.80 0.85 0.90 0.95 0.98
```

With them we do following fitting (same code as above):

```
qcurves <-
  llply( qs, function(x) {
    fit <- rq( Mean_TemperatureF ~ bs(Index, df = nKnots + 3, degree = 3), tau = x, data = tempDF)
    X %*% fit$coef
    }, .progress = "none")
qfitDF <- do.call(cbind, qcurves )
qfitDF <- data.frame(Index=1:nrow(qfitDF), Date = tempDF$Date, qfitDF )
```

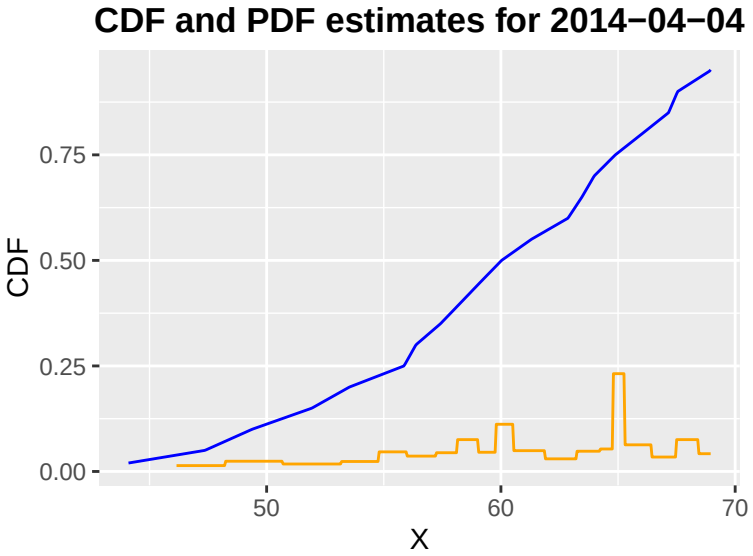
CDF and PDF re-construction

At this point we are ready to do the reconstruction of CDF and PDF for selected dates and plot them.

```
ind <- 1100
qvals <- as.numeric(qfitDF[ind, 3:(2+length(qs))]); names(qvals) <- NULL
cdfFunc <- CDFEstimateFunction( qs, qvals )

xs <- seq(min(qvals),max(qvals),0.05)
print(
  ggplot( ldply( xs, function(x) data.frame( X = x, CDF = cdfFunc(x), PDF = pdfFunc(x) ) ) ) +
    geom_line( aes( x = X, y = CDF ), color = "blue" ) +
    geom_line( aes( x = X, y = PDF ), color = "orange" ) +
    ggtitle( paste( "CDF and PDF estimates for", qfitDF[ind, "Date"] ) ) +
    theme(plot.title = element_text(lineheight=.8, face="bold"))
)
```

```
## Warning: Removed 41 rows containing missing values (geom_path).
```



References

[1] Roger Koenker, "Quantile regression in R: a vignette", (2015), CRAN.

[2] Ram Narasimhan, "weatherData: An R package that fetches Weather data from websites", http://ram-n.github.io/weatherData/.