Energy Informatics

RESEARCH Open Access

Time series analysis with apache spark and its applications to energy informatics

Cornelia Krome* and Volker Sander

From The 7th DACH+ Conference on Energy Informatics
Oldenburg, Germany. 11-12 October 2018

* Correspondence: krome@fh-aachen.de
Faculty of Medical Engineering and Technomathematics, Fachhochschule Aachen, 52428 Jülich, Germany

Abstract

In energy economy forecasts of different time series are rudimentary. In this study, a prediction for the German day-ahead spot market is created with *Apache Spark* and *R*. It is just an example for many different applications in virtual power plant environments. Other examples of use as intraday price processes, load processes of machines or electric vehicles, real time energy loads of photovoltaic systems and many more time series need to be analysed and predicted.
This work gives a short introduction into the project where this study is settled. It describes the time series methods that are used in energy industry for forecasts shortly. As programming technique *Apache Spark*, which is a strong cluster computing technology, is utilised. Today, single time series can be predicted. The focus of this work is on developing a method to parallel forecasting, to process multiple time series simultaneously with *R* and *Apache Spark*.

Keywords: Time series, Apache spark, Energy informatics, Forecast, ARMA

Content

At *EPEX Spot* exchange electricity is traded and exchange members can offer bids for buying and selling energy (EPEX Spot SE, 2018). In Germany bids for the next day must be submitted until noon. Normally, single hours are traded at the day-ahead market. The *Market Clearing Price* reveals as intersection between offered and demanded amount for each hour. In the end, every market member pays this price for a particular hour (next-Kraftwerke, 2018). The price process for January to June 2018 can be seen in Fig. 1.

In context of the *EFRE.NRW* funded project *KundenoRientiert Flexibilisierungs-poTenziale erschließen* (KRaFT) we are interested in price and load profile forecasting. Today, most platforms focus on a single time series. When applying the forecast of time series to more sources, we are facing big data problems. One example of use may be the analysis of load profiles for each electric vehicle load station in a car park. So, strategies for best loading can be applied and may

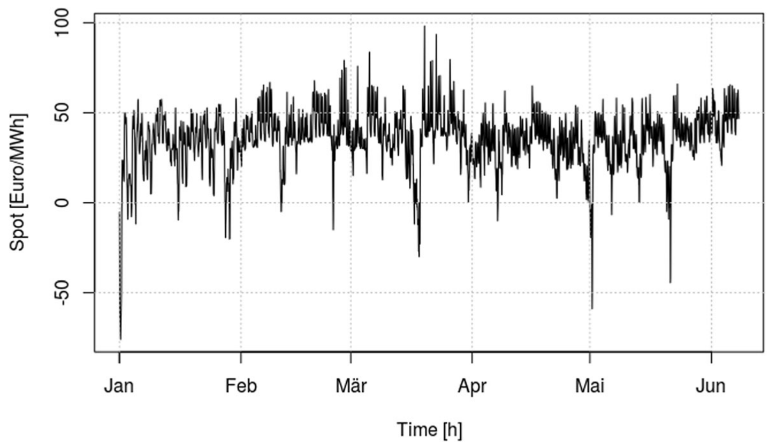

Fig. 1 Time series of German day-ahead prices in 2018

maximise profit. At the moment, the given time series can be analysed and predicted with *R* (R Core Team, 2018). But for new use cases *R* itself is not sufficient.

In this work it is analysed, how multiple time series can be processed. Therefore, parallel computing with *R* and the clustering tool *Apache Spark* (Apache, 2018a) is used. We analyse the feasibility of computing those to make differentiated forecasts for several datasets in parallel, allowing complex forecast scenarios in a virtual power plant environment as it is addressed by the *KRaFT* project.

Time series method

This section provides a short summary of *the autoregressive-moving average (ARMA)* models, which are used to predict day-ahead prices.

ARMA-models are a mix of *autoregressive (AR)* and *moving average (MA)* models. AR-models relate the current value $\tilde{x}$ of a process to a finite, linear combination of previous values of the process and a random noise ω. They get abbreviated as $AR(p)$, where p describes the order. Instead, MA-models represent $\tilde{x}$ linearly dependent on a finite number q of previous random noise ω's (Box et al., 2008).

With a mix of both model types more flexibility is achieved. This leads to ARMA models of order p and q, $ARMA(p,q)$ (Box et al., 2008).

$$\tilde{x}_t = \underbrace{\sum_{i=1}^{p} \varphi_i \tilde{x}_{t-i}}_{\text{AR part}} + \underbrace{\sum_{j=1}^{q} \theta_j \omega_{t-j}}_{\text{MA part}}$$

where φ_i and θ_j are the model coefficients.

Apache spark

With *Apache Spark* fast and general-purpose cluster computing can be performed. It can run computations in memory and is recommended for a usage with big data which are analysed parallel. The three main components of *Apache Spark* are *Spark Core* with the basic functionality, *Spark SQL* for working with structured data and *Spark Streaming* to process live streams of data (Karau, 2015) such as day-ahead prices from *EPEX Spot* or real-time energy loads of photovoltaic systems and electric vehicle load stations.

At the moment, the used architecture saves daily processed day-ahead prices from *ENTSO-E Transparency Platform* (ENTSO-E Transparency Platform, 2018) in a *Hadoop Distributed File System (HDFS)* on a cluster where Spark runs standalone. Within each time series analysis, the data is imported as HDFS-file and parallel predicted with Spark. The output can be stored as HDFS-files or with databases as *Apache Cassandra (Apache,* 2018b*)*, which provides better perspectives for the distribution of multiple time series among the cluster.

In the following sections we analyse an energy price time series with *R* and execute it on a Spark-Cluster in parallel. To use *Apache Spark* a session is needed. In *R* it is started with the following commands:

Listing 1 starting a spark-session

```
library(SparkR,lib.loc = c(file.path(Sys.getenv(”SPARK_HOME”),”R”,”lib”)))
sparkR.session(master=”spark://master:7077”)
```

Time series analysis of energy prices

The data for this analysis are taken from (ENTSO-E Transparency Platform, 2018). A script stores the day-ahead prices on the cluster. The time series of the price process can be seen in Fig. 1.

To compute a forecast, a Spark-session is started first. Subsequently, the day-ahead prices of the German energy market get imported as HDFS-file. Then the time series is created. Here the values for 2018 until June 7th are used. The dataset is divided into training and test data (the last day), which gets predicted.

There is no time series package for *Apache Spark* and *R (R Core Team,* 2018*)* on the market. So, another method to perform forecasts with *R* on a Spark-Cluster is needed. With the API *SparkR* the function spark.lapply () (Apache, 2018c) is introduced. Similar to known *R*-functions as *apply*, *sapply* and *lapply*, it runs a user defined function over a list of elements. For each element the R driver sends the function to an R worker, executes it and returns the result of all workers as a list to the R driver. With this possibility a forecast logic is written in *R* and is parallel performed for multiple time series on a Spark-Cluster. The command for running this is shown in Listing 2.

Listing 2 command for spark.Lapply

```
fcast <- spark.lapply(liste,getForecast)
```

Multiple time series, which are available as distributed datasets in the cluster, are included in liste. The user defined function getForecast (see Listing 3) processes the prediction for each element in liste.

Listing 3 method used in lapply

```
getForecast <- function(x){

        require(forecast)

        fit <- auto.arima(x)

        return(forecast(fit, h=day, level = 0.975))

}
```

All required packages need to be loaded within the function. The model of the time series is created with *auto.arima*, a known *R*-function (Hyndma & Khandakar, 2008). In this case, the time series of the day-ahead prices is modelled with an *ARMA (3, 2) with non-zero mean* model. With the *R*-function forecast (Hyndma & Khandakar, 2008) a prediction of the next 24 values (1 day) is calculated based on the training process.

In Fig. 2 the forecast (red) is shown. Additionally, the historic values of 1 week (black), the real observations for the last day (green) and the confidence interval for the prediction (blue) is plotted.

Summary

To submit bids for electricity at the *EPEX Spot* a good knowledge of needs and a price forecast is recommended. Therefore, different methods can be used, for instance a simple *ARMA*-model can predict the first few hours of a day. Other models and a preceded clustering of the data may bring a better result.

The first test of *Apache Spark* and *R* showed, that even if there is no special package for time series analysis in *R* with *Apache Spark*, it is possible to run parallel forecasts on a Spark-Cluster in *R*. So, multiple time series of price processes or load profiles like those of electric vehicle load stations in a car park, can be done simultaneously.

In future works, the scalability of time series analyses with *Apache Spark* and *R* with the developed method will be thematised.

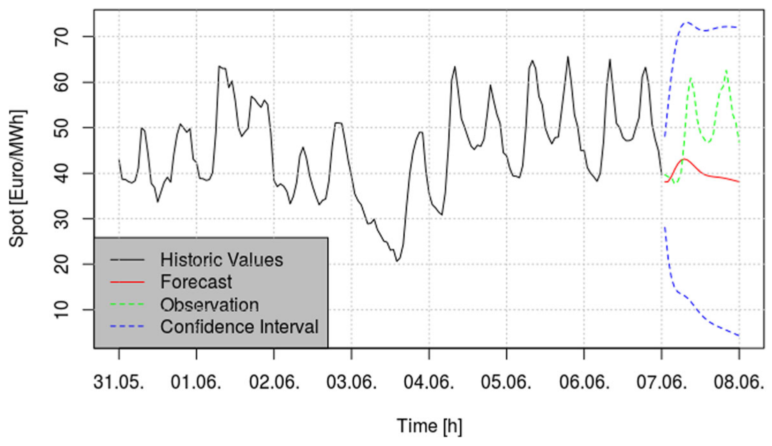

Fig. 2 Forecast of day-ahead prices of June 7th 2018

Abbreviations
AR: Autoregressive; ARMA: Autoregressive-moving average; HDFS: Hadoop Distributed File System; KRaFT: KundenoRientiert FlexibilisierungspoTenziale erschließen; MA: Moving average

Funding
Publication of this article was sponsored by *EFRE.NRW* grant.
Publication costs for this article were sponsored by the Smart Energy Showcases - Digital Agenda for the Energy Transition (SINTEG) programme.

Availability of data and materials
Datasets related to this article can be found at https://transparency.entsoe.eu/transmission-domain/r2/dayAheadPrices/show, with area *Germany, BZN|DE-AT-LU* and year 2018 (ENTSO-E Transparency Platform, 2018).

About this supplement
This article has been published as part of *Energy Informatics* Volume 1 Supplement 1, 2018: Proceedings of the 7th DACH+ Conference on Energy Informatics. The full contents of the supplement are available online at https://energyinformatics.springeropen.com/articles/supplements/volume-1-supplement-1.

Authors' contributions
Both CK and VS contributed to the final version of the manuscript. Both authors read and approved the final manuscript.

Competing interests
The authors declare that they have competing interests as they are working on a project called *KundenoRientiert FlexibilisierungspoTenziale erschließen* which is sponsored by *EFRE.NRW*.

Publisher's Note

Published: 10 October 2018

References
Apache. Apache Spark - Unified Analytics Engine for Big Data. [Online] 17. 07 2018a. https://spark.apache.org/
Apache. Apache Cassandra. [Online] 17. 07 2018b. http://cassandra.apache.org/
Apache. R: Run a function over a list of elements, distributing the... [Online] 07. 06 2018c. https://spark.apache.org/docs/2.0.2/api/R/spark.lapply.html
Box GE, Jenkins P, Gwilym M, Reinsel GC (2008) Time Series Analysis - Forecasting and Control, vol 4. John Wiley & Sons, Inc., New Jersey
ENTSO-E Transparency Platform. ENTSO-E Transparency Platform. [Online] 19. 06 2018. https://transparency.entsoe.eu/
EPEX Spot SE. About EPEX SPOT. [Online] 19. 06 2018. http://www.epexspot.com/en/company-info/about_epex_spot
Hyndma RJ, Khandakar Y (2008) Automatic time series forecasting: the forecast package for {R}. J Stat Softw 26:1–22
Karau H et al (2015) Learning Spark: lightning-fast data analysis. O'Reilly, Beijing
next-Kraftwerke. Day-Ahead-Handel - Was Ist Das? [Online] 19. 06 2018. https://www.next-kraftwerke.de/wissen/strommarkt/day-ahead-handel
R Core Team. R: a language and environment for statistical computing. [Online] 2018. https://CRAN.R-project.org/package=sparklyr