

Providing a user-friendly outlier analysis service implemented as open REST API

Doron Goldfarb*, Johannes Kobler, Johannes Peterseil

Umweltbundesamt GmbH, Vienna, Austria

* Corresponding author: doron.goldfarb@umweltbundesamt.at

EGU 2020, Virtual Meeting

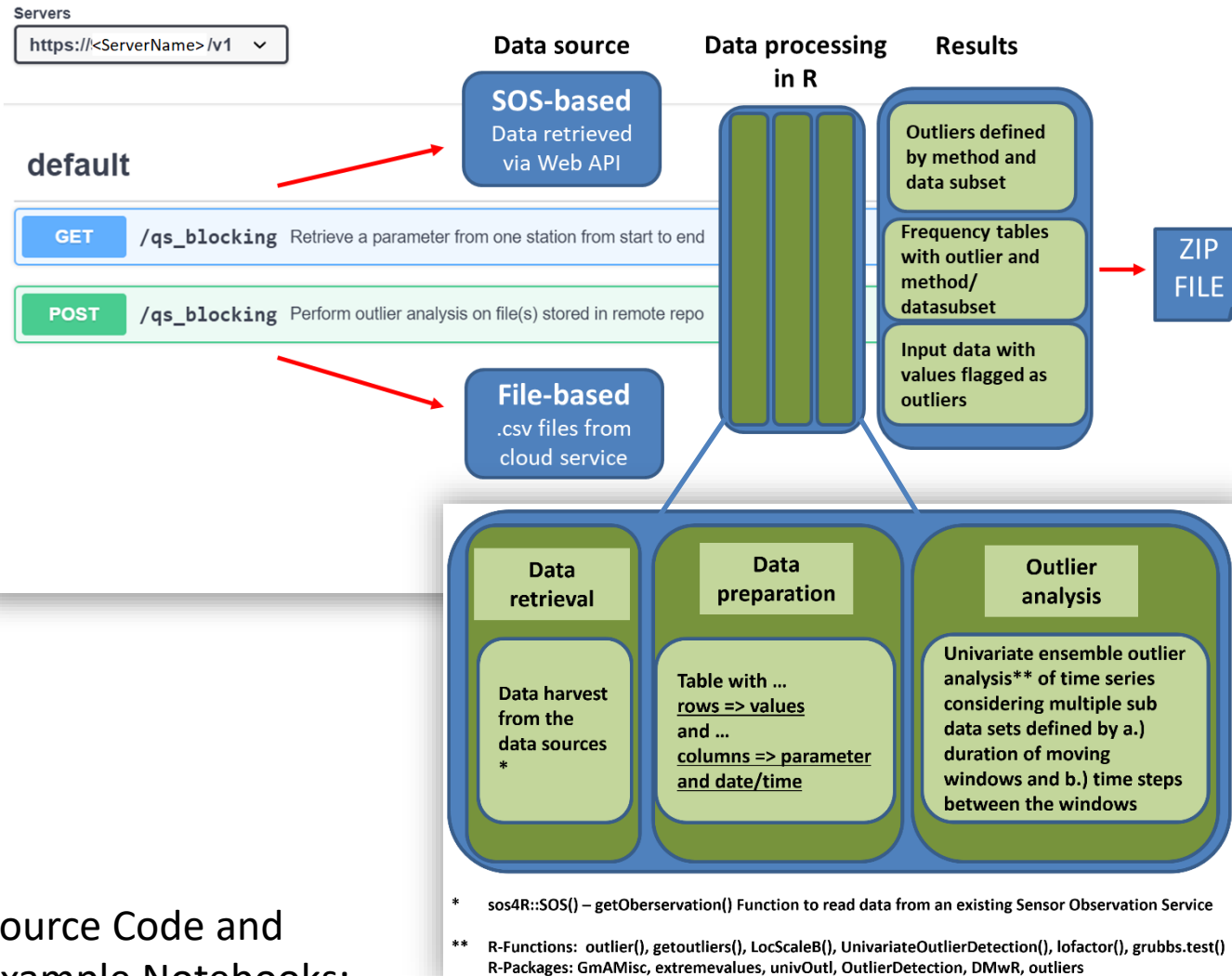
This work was done in the context of the EOSC-HUB (grant agreement no 777536) project which has received funding from the European Union's Horizon2020 research and innovation programme.



Outline

- Outlier detection is crucial to reliable research
- Definition of „outlier“ varies by application
- Increasing online availability of environmental time-series data
- Approach: Provide public Web service for outlier analysis
 - Based on ensemble of outlier detection methods from different R-packages
 - Operating on data sourced from Sensor Observation Service (SOS)
or from files from the cloud
 - Accessible via REST API
- Outlook: Refine initial EOSC-HUB prototype throughout eLTER+ project

Timeseries Outlier Analysis ^{0.1} OAS3



JupyterLab

notebooks.egi.eu/user/f9f035fc4b4971ae2c1c06c96...

```
[1]:
library(ggplot2)
library(reshape2)
library(httr)

request_url_block_sos="https://96.147.182.53:8080/v1/qs_blocking"

query_block_sos_list(
  sosendpoint="https://sensorweb.demo.52north.org/sensorwebtestbed/service",
  parameter="AirTemperature",
  site="ELV-W52500",
  begin="2015-08-02T00:00:00.000Z",
  end="2015-08-17T23:59:59.000Z",
  usecache=FALSE,
  windowed=TRUE,
  windowinterval=3
)

request_url=request_url_block_sos
query=query_block_sos_list

set_config( config[ ssl_verifypeer = 0L, ssl_verifyhost = 0L ] )
res=GET(request_url, query=query)

zipfile=tempFile(fileext=".zip")
file=file(zipfile, "w")
writeBin(content(res), file)
close(file)

results=utils::unzip(zipfile, list=TRUE)
files=results$Name[grepl("total_timeseries_outlier.dat", results$Name)]
ts=read.table(unz(zipfile,files), sep = ",", header=T)

ts$ts=ts$FILENAME[,""]
ts$FILENAME=Factor(ts$FILENAME)

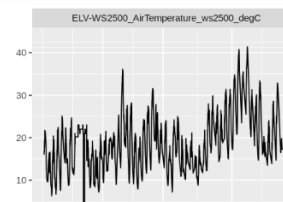
ts$timestamp=as.POSIXct(substr(ts$ts, 1, 19), as.character(ts$timestamp), format = "%d-%m-%y %H:%M:%S", tz="UTC")

ts=ts[order(ts$timestamp),]
ts_outliers=ts[ts[,c(3,4,5,6,7,8,9,10)], id.vars=c("timestamp", "FILENAME")]
ts_outliers$ts_outliers[is.na(ts_outliers$value) & ts_outliers$value > 0,]

ts_outliers$variable=Factor(ts_outliers$variable)
minval=min(ts$value, na.rm=T)
maxval=max(ts$value, na.rm=T)
diff=maxval-minval
difflog=log(diff)
rounder=round(difflog)
interval=round(difflog/rounder)*rounder
minlab=floor(minval/interval)*interval
maxlab=ceiling(maxval/interval)*interval

fact=maxlab/length(levels(ts_outliers$variable))
y_breaks=c(unique(as.numeric(ts_outliers$variable))-(1:length(levels(ts_outliers$variable))))*fact, seq(minlab, maxlab, interval))
y_labels=c(levels(ts_outliers$variable), seq(minlab, maxlab, interval))

ggplot(ts, aes(x=timestamp)) +
  facet_grid(.~FILENAME) +
  geom_line(aes(y=value, group=FILENAME)) +
  geom_point(data=ts_outliers, na.rm=TRUE, aes(y=(as.numeric(variable)-(1:length(levels(variable))))*fact, color=factor(value)), size=0.1) +
  scale_color_manual(values=c("550000", "FF0000")) +
  scale_y_continuous("", breaks=y_breaks, labels=y_labels) +
  scale_x_datetime(date_labels="%d-%m-%y %H:%M:%S")
  theme(axis.text.x=element_text(angle = 45, hjust=0.95))
```



factor(value)

1

2



Source Code and
 Example Notebooks:

<https://github.com/d0rg0ld/OutlierDetection4EOSC>