National Research Programme „Cyber-physical systems, ontologies and biophotonics for safe&smart city and society.” (SOPHIS)

Project No.4. „Development Development of technologies for secure and reliable smart-city”

„DRINKING WATER QUALITY MONITORING AND CONTAMINATION DETECTION SYSTEM”

Technology and algorithm description

Riga Technical University
Water Research Laboratory

ELEKTRONIKAS UN DATORZINĀTNU INSTITŪTS INSTITUTE OF ELECTRONICS AND COMPUTER SCIENCE

LATVIJAS UNIVERSITĀTE ANNO 1919

1862 RĪGAS TEHNISKĀ UNIVERSITĀTE

Latvijas Universitātes Matemātikas un informātikas institūts

Contents

1. Introduction

Drinking water (DW) supply systems are vulnerable to deliberate and accidental water contamination events. Modern drinking water DW quality monitoring methods and accessibility to them have indicated that DW quality contamination events are still an issue all over the world.

On-line DW quality monitoring and early warning systems (EWS) were designed to provide remote and continuous DW monitoring and to notify the water utilities and consumers about the contamination events in DW supply system, and therefore to prevent possible contaminant exposure (Storey et al., 2011). EWS consists of a drinking water quality sensors set, data collection and analysis system and alarm triggering algorithm (Liu et al. 2016; Liu, et al. 2015; Liu et al. 2014).

The measurement tools usually consist of non-specific compound sensors, such as temperature, oxidation-reduction potential (ORP), pH, conductivity, etc. (Jeffrey Yang et al., 2009; Liu et al., 2015). Advantages of non-compound specific sensors are their accuracy, relative simplicity, low installation and operational costs. Moreover, it is not possible to predict the type of contamination, which could occur in the system and install compound specific sensors. Thus precision and fast response of the non-compound specific sensors is of higher importance.

A critical factor for properly working EWS is the detection algorithm (Liu et al., 2014). Mathematical algorithms were developed through the decades to recognize the contamination events between normal periodic fluctuations of drinking water quality.

2. Overall system description

Drinking water quality monitoring and contamination system consist of two parts – the monitoring section at the site, and the data processing machines at the data center. Data from the site to the data center is transferred via the internet, powerlines (BPL technologies) or telemetry (figure 1). The overall look of the system is shown in figure 2. Data is collected at the drinking water quality monitoring point in the drinking water distribution system with a time step of 1 minute. At each time step, the data from all sensors are packed as one set of parameters and transferred to the *data center*. The data center is the computing system where measurement data is equalized (signal noise reduction), processed and following the alarm is triggered in a case of contaminant detection.

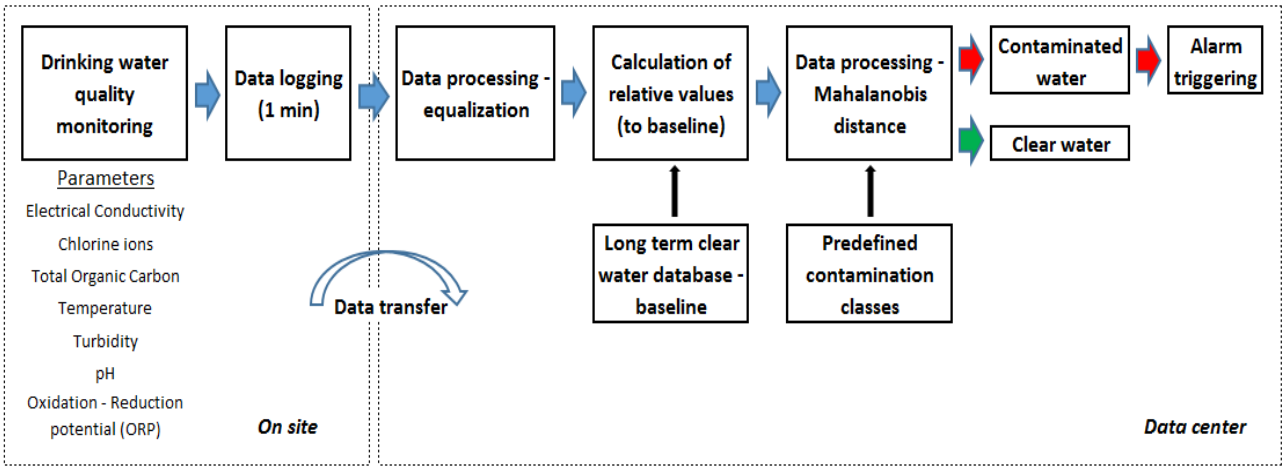

Figure 1. The drinking water quality monitoring system – principle scheme

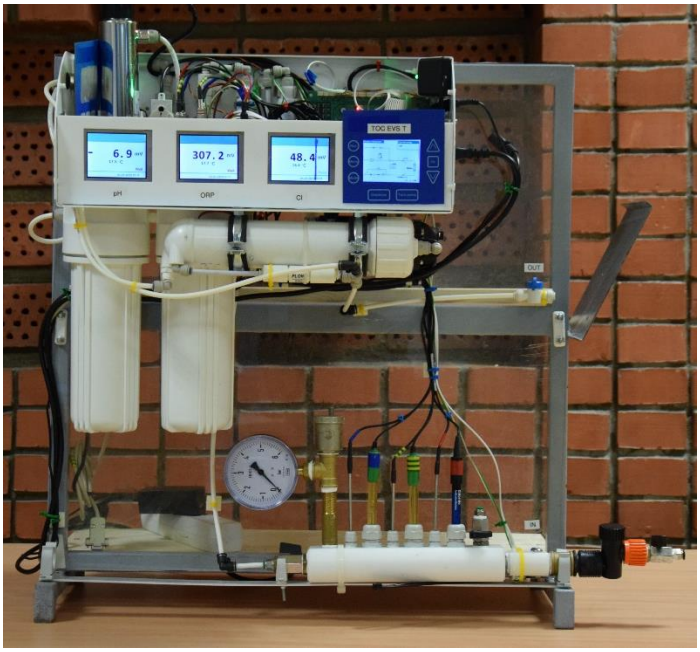

Figure 2. The drinking water quality monitoring system

2.1. On-site equipment

A specific set of DW quality monitoring sensors is selected and is the core part of the monitoring system. The selected parameters and sensors are summarized in table 1. The sensors are installed in special bindings (figure 3). The DW parameters are measured with a time step of 1 minute. When the data from all sensors is collected, it is packed in one file and sent to the data center for further data processing. Also, the measurements are visualized by LED screens an can be observed manually.

To maintain the stability of readings and contamination detection, sensors should be calibrated and maintained once a week. The calibration and maintenance must be done accordingly to manufacturer instructions for each sensor. During every maintenance event (1 week), the ion exchange resin and ultrafiltration membranes for TOC sensors should be checked and in a case of necessity replaced (figure 4).

Table 1.

Drinking water quality sensors

Parameter	Sensor	Range
Electrical conductivity (EC)	EVS Adrona Ltd. (Latvia)	0 – 2000
Chloride ions (Cl^-)	Ion Selective Electrode (ISE) K-27502-13 (Cole-Parmer Ltd.)	0 – 3550 mg/l
Total organic carbon (TOC)	TOC Adrona Ltd. (Latvia) – UV oxidation	0 – 30 mg/l
Temperature	T Adrona Ltd. (Latvia)	0 – 30 ^oC
pH	Combination pH electrode HI1230B (Hanna Instruments Ltd.)	0 – 12
Turbidity	TurbiGuard (SIGRIST-PHOTOMETER AG)	0 – 4000 NTU
Oxidation – reduction potential (ORP)	ORP electrode HI3230B (Hanna Instruments Ltd.)	-2000 – 2000 mV

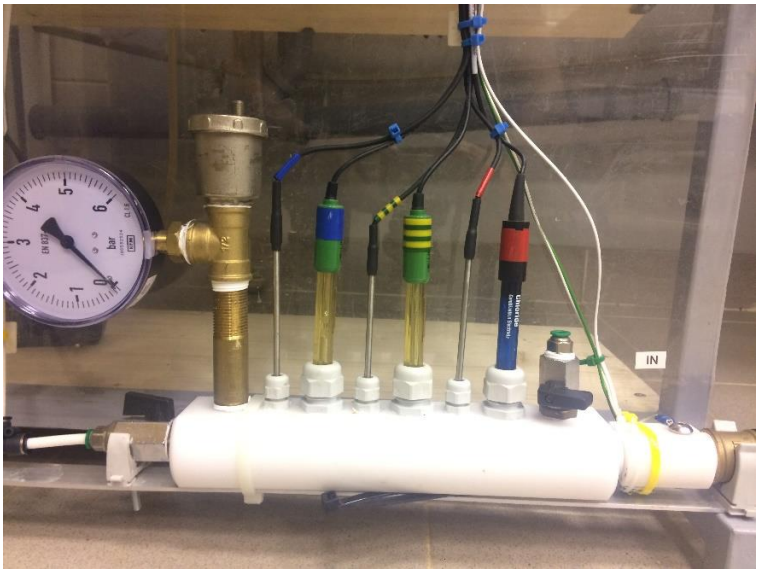

Figure 3. Set of drinking water quality monitoring sensors

With this configuration, the system should be installed in dry and warm place to avoid the freezing of system and electrical failures due to water intrusion in it.

Figure 4. Ultrafiltration membrane and ion exchange resin for TOC measurements

Technical data of system is shown in table 2.

Table 2.

Technical data

Name	Data
Power supply	220 - 230 V
Emergency power source	UPS Energenie (60 min – 1500 VA)
Data collection unit	8 x Adrona AD2003_Analog v2.4A
LCD screens	4
Data connection (transfer)	2 x USB 2.0
Data center	Intel® Core ™ i3-6000 CPU @ 3,70 GHz, RAM 4,00 GB
Inflow connection (water)	8 mm
Outflow connection (water)	8 mm
Max flow	10 l/h
Max pressure (inflow)	1 bar
Pressure reducing valve	Included (max 1 bar)
Ultrafiltration pump	Diaphragm pump AQ&Q E150-ABP-1024, 24 V, 2,2 l/min
Ultrafiltration membrane	AXEON TF – 1812 – 100 (4,5 bar, 15,77 l/h)
Ion exchange resin	Amberlite MB20
Dimensions	70 x 70 x 45 mm
Weight	22 kg

2.2. Data center

The received data at the data center is proceeded with contamination detection algorithm to ensure the contamination detection. After the first installation of drinking water quality monitoring and contamination detection system, the long-term monitoring should be accomplished to define the drinking water quality baseline, which represents the DW quality changes during the normal conditions and clear water in drinking water supply system. Long-term monitoring must be done for at least 30 days to collect enough data to define a baseline. After the long-term monitoring a *CLEAR WATER* class can be set in the contamination detection algorithm as an input data.

Potential contamination agent classes *Wastewater*, *Groundwater*, *Surface water* and *E.coli* are previously defined and included in the algorithm as default values. Since the *Clear Water* class parameter values varies from one network to other because of various water sources and quality level the total raw reading values of parameters cannot be used. Due to DW quality variations the contamination classification classes are set as set as relative values *RBx* to baseline in various and independent DW supply systems. The calculation of *RBx* is shown in chapter 3. Each of contaminant classification class consists of 6 parameters and 40 repetitions that makes a total number of 240 readings. Average *RBx* values are summarized in table 3.

Table 3.

RBx values for various drinking water contaminants

	EC	**TOC**	**pH**	**Temperature**	**ORP**	**Turbidity**
Clear water	1,00	1,00	1,00	1,00	1,00	1,00
Surface water	1,05	1,16	1,01	1,03	1,03	2,53
Groundwater	1,79	1,18	1,02	1,03	0,98	4,48
Wastewater	1,26	1,79	1,01	1,03	0,07	9,92
E.coli	1,03	1,22	1,00	1,02	1,03	1,13

After the definition of all classes, the system can be started as a contamination detection tool in drinking water supply system. At each time step, the results are compared to previously set classes, and each measurement is classified as a clear or contaminated water. The raw readings from sensors, RBx values, classification results and decision on water quality is automatically stored in output file and the alarm is triggered in a case of contamination event.

The example of collected raw data from sensors during contamination events (3 repetitions, every 2 hours) is shown in figure 5.

The example data after calculation of relative values for all parameters during contamination events is shown in figure 6.

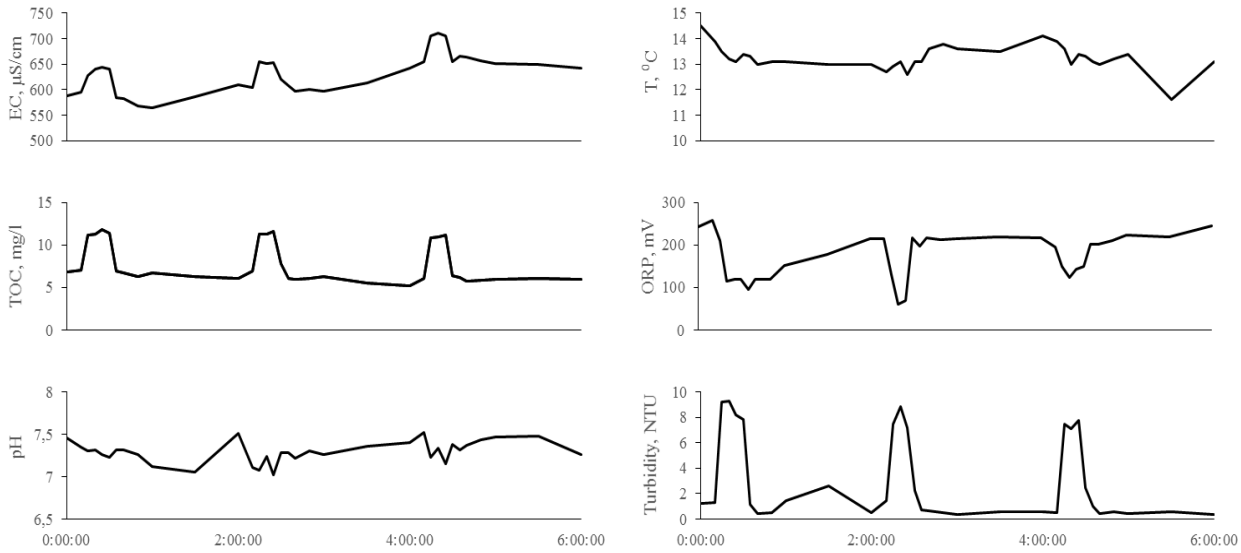

Figure 5. Raw data from drinking water quality monitoring sensors

The contamination detection algorithm is described in chapter 3. If the drinking water is classified as a contaminated water, the alarm is triggered, and water utilities must apply the action that is described in corresponding Water safety plan for the DW supply system. Depending on the potential consequences of contamination event, the various types of actions could be applied, starting from warning the consumers about low-quality water that should not be used as a drinking water, up to shutting down the system completely and cleaning/disinfection of DW supply system.

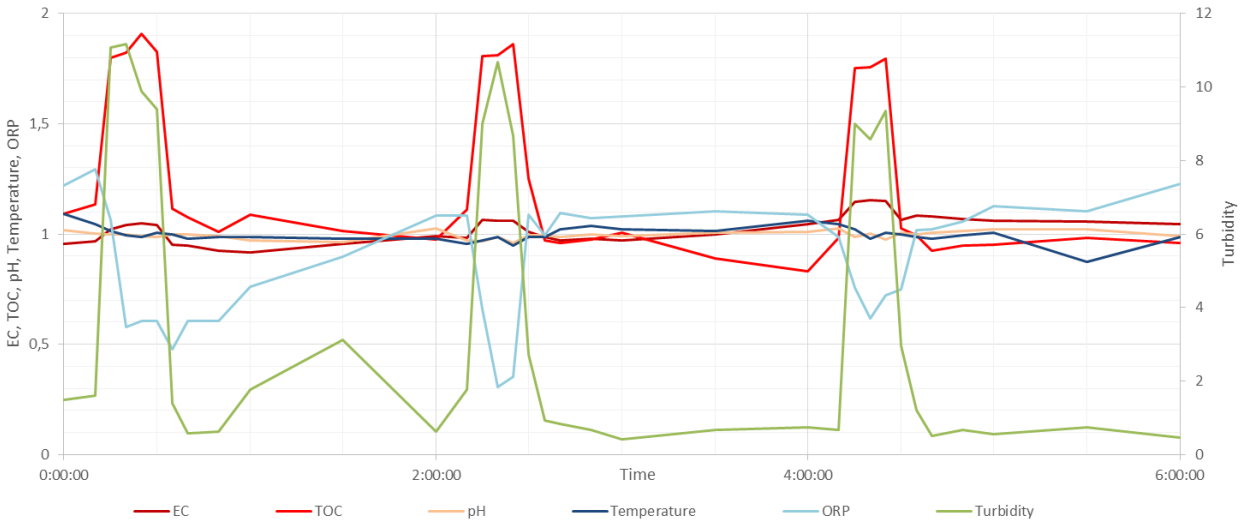

Figure 6. Sensor data after the calculation of relative variations (*RBx)* during contamination events

3. Contamination detection algorithm

The method is based on the assumption that similar objects have close values for at least a set of dimensions (Liu et al., 2015a). If the distance from an object is shorter than the distances to other classes, then the object is deemed belonging to that class. In this study, 2 classes were defined: clean water, which parameters were typical for water without contamination (drinking water), and contaminated water.

For the convenience of data comparison, the absolute values of the parameters were expressed as relative changes of each parameter:

$$RB_x = \frac{R_x}{B_x} \tag{1}$$

where Rx is the raw reading or measurement of parameter x, Bx is the average value of baseline for parameter x, and RBx represents the relative change of sensor against its baseline

Also the fingerprints of each contaminant have been determinate and represented as clean (uncontaminated) and contaminated water ratio, and shown as a multidimensional spider graphs on figure 7.

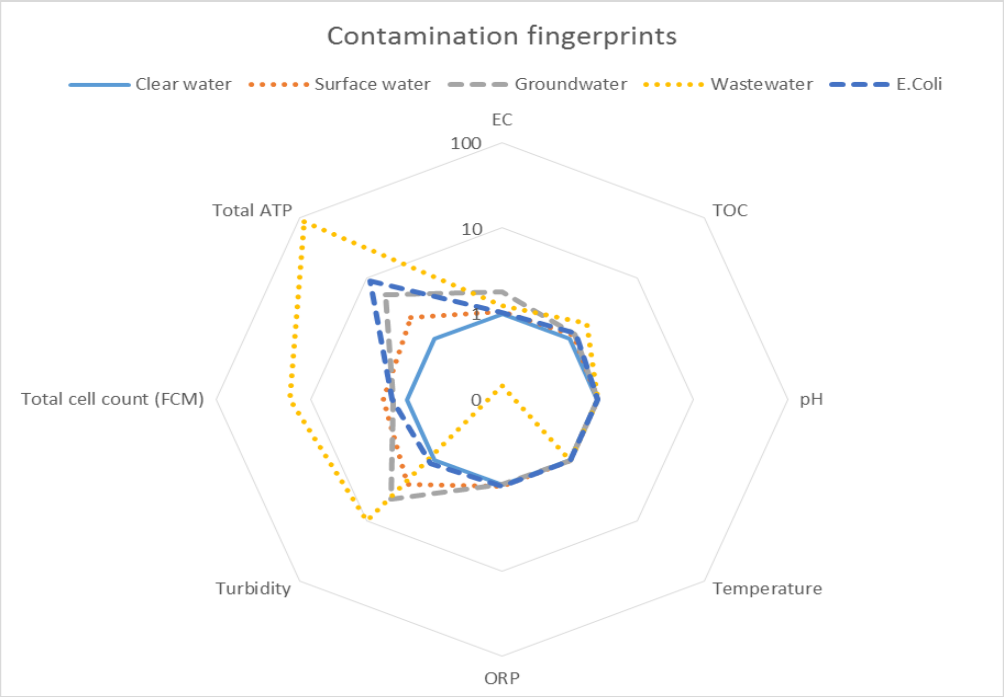

Figure 7. Fingerprints of various contaminants

A set of the parameters, measured for the sample, is defined as one object in multidimensional space, where a number of dimensions corresponding to the quantity of parameters and the scale is normalized relatively to all parameters. The distance from the current object to previously set class (point in multidimensional space) is calculated. If the distance from an object is shorter than the distances to other classes, then the object is deemed belonging to that class (Liu et al., 2015). Mahalanobis distance can be used to identify and gauge similarity of unknown sample set (object) to a known one (class). If object p =

(p1,p2, …, pn) and class c = (μc1, μc2, …, μcn), where μc is the mean of all instances in class c, are points in n-space, then the Mahalanobis distance from p to c, or from c to p is given by

$$D_M(p,c) = \sqrt{(p-c)^T S^{-1}(p-c)} \quad (1)$$

where DM is the Mahalanobis distance, S is the covariance matrix of q, T is transposer. The distance to previously assumed and set amount of classes is calculated. It is deemed that the object p belongs to the class with the shortest distance DM to it.

The contamination detection algorithm is introduced in Matlab R2016 software and the source code is shown in appendixes 1,2 and 3.

4. Conclusions

Based on high interest and topicality of safe drinking water supply, a drinking water quality monitoring and contamination detection system have been developed. The system has been verified and tested for various contaminants. The system and proposed algorithm (Mahalanobis distances) are capable of detecting groundwater intrusion, failure of the drinking water treatment plant, wastewater intrusion in drinking water supply system due to cross-connections.

The limitations of the developed system are the pressure resistance – max 1 bar. It means that the system should be maintained with a high precision because of possible damage caused by pressure leakage from drinking water supply system.

Future studies and experiments with different pollutants, contaminant concentrations, flows and other physical and chemical properties should be done to define different contamination types, classes and determine the limits of the detection algorithm and monitoring system.

5. References

Jeffrey Yang, Y., Haught, R.C., Goodrich, J.A., 2009. Real-time contaminant detection and classification in a drinking water pipe using conventional water quality sensors: Techniques and experimental results. J. Environ. Manage. 90, 2494–2506.

Liu, S., Che, H., Smith, K., Chen, L., 2014. Contamination event detection using multiple types of conventional water quality sensors in source water. Environ. Sci. Process. Impacts 16, 2028–2038.

Liu, S., Che, H., Smith, K., Chang, T., 2015. A real time method of contaminant classification using conventional water quality sensors. J. Environ. Manage. 154, 13–21.

Liu, S., Li, R., Smith, K., Che, H., 2016. Why conventional detection methods fail in identifying the existence of contamination events. Water Res. 93, 222–229.

Storey, M., V., van der Gaag, B., Burns, B., P., (2011) Advances in on-line drinking water quality monitoring and early warning systems. Water Research 45(2) (2011), 741-747

Appendix 1

Source code – Signal processing and contamination detection

```
% sagatavo ūdens klašu parametrus un nosaka katra mērījuma klasi
nklases = 5;    % klašu skaits
parSkaits = 8;  % parametru skaits

failuVaardi = cell(1,nklases);
% te jāieraksta īstie failu nosaukumi, visiem CSV failiem jābūt darba mapē
failuVaardi{1} = 'TIRS_UPE_GRUNTS_WASTE_ECOLI_OAB_2_FA.csv';
failuVaardi{2} = 'UPE_UPE_AB_2_FA.csv';
failuVaardi{3} = 'GRUNTS_GRUNTS_AB_2_FA.csv';
failuVaardi{4} = 'WASTE_WASTE_AB_2_FA.csv';
failuVaardi{5} = 'ECOLI_ECOLI_AB_2_FA.csv';
%failuVaardi{6} = 'netirs5.csv';
% ... tik failu vārdu, cik ir klases

klasuVaardi = cell(1,nklases);
% te jāieraksta īstie attiecīgo klašu nosaukumi
klasuVaardi{1} = 'CLEAR';
klasuVaardi{2} = 'RIVER';
klasuVaardi{3} = 'GROUND';
klasuVaardi{4} = 'WASTE';
klasuVaardi{5} = 'ECOLI';
%klasuVaardi{6} = 'Netīrs 5';
% ... tik failu vārdu, cik ir klases

% << 2017-03-17 IM lasa maksimālās distances visām klasēm (1 rinda, secība kā
'klasuVaardi'
try
    maxDist = csvread('MaxDist5.csv');
catch
    error ('Maksimālo distanču faila lasīšanas kļūda');
end
[rows, cols] = size(maxDist);
if rows~=1 || cols~=length(klasuVaardi), error ('Nepareizs maksimālo distanču fails!');
end
% >> 2017-03-17 IM

klases = struct('centrs',[],'kov1',[],'det',[]);
% lasa klašu (klasteru) datus no CSV failiem un izveido klasterus
for iklase = 1:nklases
    try
        klasesDati = csvread(failuVaardi{iklase});
        [merSkaits, parSkaitsKlasei] = size(klasesDati);
        if parSkaitsKlasei~=parSkaits,
            error (['PAR> ' num2str(iklase) '.klasei nepareizs parametru skaits']);
        end
    catch ex
        msg = ex.message;
        if strcmp(msg(1:3), 'PAR'), error (msg (6:length(msg)));
```

```
        else error (['Neatradu klases datu failu: ' failuVaardi{iklase}]);
        end
    end
    klases(iklase) = klasteraParametri(klasesDati);
    if klases(iklase).det == 0,
        error ([num2str(iklase) '.klasei kovariāciju matrica singulāra']);
    end
end

% lasa mērījumu datus
try
    merijumi = csvread('NOLASIJUMI_UPE_GRUNTS_WASTE_ECOLI_OAB_II_III_2_FA.csv');
    [merSkaits, parSkaitsMer] = size(merijumi);
    if parSkaitsMer~=parSkaits,
        error ('PAR> Mērījumos nepareizs parametru skaits');
    end
catch ex
    msg = ex.message;
    if strcmp(msg(1:3), 'PAR'), error (msg (6:length(msg)));
    else error (['Neatradu mērījumu failu: ' failuVaardi{iklase}]);
    end
end

% katram mērījumam atrod tuvāko klasi un ieraksta visas distances
merijumuKlases = zeros(1,merSkaits);
merijumuDistancesLidzKlasem = zeros(merSkaits, nklases);
for imer = 1:merSkaits
    [merijumuKlases(imer), merijumuDistancesLidzKlasem(imer,:)] =
tuvKlasteris(merijumi(imer,:), klases);
    dd = merijumuDistancesLidzKlasem(imer,:) - maxDist;
    % << 2017-03-17 IM
    if all(dd>0),
        disp ([num2str(imer) ':UNKNOWN' ' '
num2str(merijumuDistancesLidzKlasem(imer,:))]);
    else
        disp ([num2str(imer) ':' klasuVaardi{merijumuKlases(imer)} ' '
num2str(merijumuDistancesLidzKlasem(imer,:))]);
    end
    % >> 2017-03-17 IM
end
```

Appendix 2

Source code – Contamination classification

```
function [klasteris, distances] = tuvKlasteris(vektors, klasteri)
% Atrod parametru vektoram tuvāko klasteri
%
% Ieejā:
%       vektors - vektors ar garumu M (parametru skaits)
%   klasteri - struktūru masīvs ar garumu K (klasteru skaits)
%       klasteri(i).centrs - i-tā klastera centrs
%       klasteri(i).kov1 - i-tā klastera invertēta kovariācijas matrica
% Izejā:
%       klasteris - tuvākā klastera indekss
%   distances - Mahalanobisa distances līdz visiem klasteriem (garums K)

x=vektors';

klasteruSkaits = length(klasteri);
distances = 1.e10 *ones(1,klasteruSkaits);
for i=1:klasteruSkaits
    centrs = klasteri(i).centrs;
    if isempty(centrs), continue; end
    distances(i)=(x-centrs)'*klasteri(i).kov1*(x-centrs);
end
klasteris=find(distances==min(distances));
```

Appendix 3

Source code – Cluster analysis

```
function klasteris = klasteraParametri(vektori)

% aprēķina klastera parametrus no tajā ietilpstošiem vektoriem
%
% Ieejā:
%       vektori - matrica NxM, kur N- klasterī ietilpstošo vektoru skaits, M- spektra
joslu skaits
% Izejā:
%   klasteris.centrs - vidējais vektors (klastera centrs)
%   klasteris.kov1 - invertēta kovariācijas matrica ([], ja neinvertējās)
%   klasteris.det - kovariācijas matricas determinants. Ja =0, matrica neinvertējās

klasteris.centrs=mean(vektori)';
kov = cov(vektori);
klasteris.kov1=inv(kov);
klasteris.det=det(kov);
```