Water Consumption Periodic Autoregressive Time Series Iterative Forecasting

Taras Mykhailovych[a], Mykhailo Fryz[a], Iaroslav Lytvynenko[a]

[a] *Ternopil 'Ivan Puluj' National Technical University, Rus'ka st., 56, Ternopil, 46001, Ukraine*

Abstract

Problems of urban water provision enterprises, such as: geographical zoning, pressure normalization, emergency detection, were stated. Water consumption of such an urban water provision enterprise is being studied by the authors. Water consumption mathematical model was presented and justified as cyclostationary. Hourly water consumption time series were justified in terms of linear forecasting. Water consumption operative interval forecasting method and its existing authors' information technology were presented and revised. Problems of existing information technology were stated. Improvements to information technology were proposed. The advantages of the iterative forecasting method were explained. Streaming algorithm of water consumption forecasting was proposed and presented. Algorithm implementation and deployment preferences were proposed. Next steps of the research were defined.

Keywords [1]

Water consumption, cyclostationary process, operative linear forecasting, time series, periodic autoregressive, prediction interval, streaming algorithm, iterative method

1. Introduction

The practical value of the urban water consumption forecasting technology is providing efficient management of clean water supply at the pressure, needed by customers [1, 2]. As per [3], an information technology, which implements hourly water consumption operative interval forecasting method, may solve the significant problems of urban water provision enterprises, such as: geographical zoning, pressure normalization, emergency detection. It may be mostly useful in scaling ahead the water pipeline systems and their emergency states detection.

Subjects of study are as following.

- Water consumption — a stochastic process: an oscillation of water flow (measured in liters per hour) in time
- Hourly water consumption — a stochastic time series: hourly water volume (measured in liters)
- Unit water consumption — one short-term consumption session (i.e. temporary valve opening) by any impartible user (i.e. a person). Example of unit water consumption is "Mr. John Smith washes his hands after returning home at 9pm, Sep 18, 2021"

First two items above describe the consumption from urban water provision enterprise in a broad sense: by some user: a person, an apartment, a building, a zone (geographic or logistic group of buildings) etc., and of any scope (any combination of consumption sessions by that user).

Object of study is operative (one step ahead) interval forecasting. Resulting prediction intervals are considered as highly likely expected water consumption bounds and are "red lines" of theoretically normal operation of the water provision system. Falling out from the prediction interval is a display of probable emergency, which may be false positive due to the human factor, yet another reason to ensure the sanity of the related system node.

[1]ITTAP'2021: 1nd International Workshop on Information Technologies: Theoretical and Applied Problems, November 16–18, 2021, Ternopil, Ukraine
EMAIL: tarquas@gmail.com (T. Mykhailovych); mykh.fryz@gmail.com (M. Fryz); d_e_l@i.ua (I. Lytvynenko)
ORCID: 0000-0002-8138-3642 (T. Mykhailovych); 0000-0002-8720-6479 (M. Fryz); 0000-0001-7311-4103 (I. Lytvynenko)

In [3] the authors described an information technology: it was developed for specific needs of an urban water provision enterprise. It allows the collection and processing of data from water consumption monitoring hardware.

The most computation-heavy part of information technology is PAR (periodic autoregressive model) parameters estimation method, which requires inversions and multiplications of very big matrices on a regular basis (every hour). This computation improves the PAR parameters estimates after acquiring each new hourly water consumption value. The updated parameter estimates are then used to compute prediction intervals for the next hours.

Recurring processing of all available historical data for PAR parameters estimation have introduced a problem: the computation became slower for each next cycle of estimation, and kept requiring more and more runtime and computing resources. This is so, because the calculation was done using all available historical data from scratch, without reusing the results from previous computations.

This paper proposes developing a method to update the PAR parameters estimates, without need to recompute them from a whole range of historical data. This will be achieved by developing an one-pass streaming algorithm for real-time PAR parameter estimation upon the iterative method of least squares updating, proposed in [4].

Another problem is current deployment of mentioned information technology on a custom dedicated server, which is not reliable, because it has following limitations.

- Requires specific computational hardware (a video card)
- Requires extreme electric energy consumption (for video card)
- Requires 24/7 unchoked operation
- Is not horizontally scalable
- Is not backed by a replica server

A new water consumption operative interval forecasting method, upon updating streaming algorithm implementation (which is low-memory and low-CPU consuming), allows to take advantage of cloud environments and move the deployment to a serverless computing platform, in order to do the following.

- Conserve the energy resources
- Conserve the costs on hardware wearout and development operations
- Scale the computations horizontally by users
- Have reliable 24/7 service and backing from the platform

An approach of deployment transition to such a platform is also proposed in this paper.

2. Related Works

Recently known water consumption models, short-term forecasting methods, and current state of research in the field were described in a digest [5] of considered publications. They were analyzed by authors in [3], in respect to stated in [3] problems of urban water provisioning on an enterprise, and are considered as incomplete for the following reasons.

- Not being stochastic
- Not being cyclostationary
- Having insufficient theoretical background

In [3], the authors defined the water consumption as an additive stochastic mathematical model of unit water consumptions. Each unit water consumption $\breve{\xi}_k(t),\ k \in \mathbb{Z}$ is approximated by scaled boxcar [6] function $\breve{\xi}_k(t) = \alpha_k \mathbf{\Pi}_{\tau_k,\, \tau_k + \widetilde{l}_k}(t)$, defined by three parameters: start time $\tau_k,\ \tau_k \in \mathbb{R}$, $\tau_m \neq \tau_n \forall\, m \neq n$, average flow $\alpha_k,\ \alpha_k \in (0, \infty)$, and duration $\widetilde{l}_k,\ \widetilde{l}_k \in (0, \infty)$. According to this, water consumption process $\xi(t),\ t \in \mathbb{R}$, is a sum of unit water consumptions:

$$\xi(t) = \sum_{k=-\infty}^{\infty} \breve{\xi}_k(t) = \sum_{k=-\infty}^{\infty} \alpha_k \mathbf{\Pi}_{\tau_k,\, \tau_k + \widetilde{l}_k}(t) = \sum_{k=-\infty}^{\infty} \alpha_k\, \phi(t - \tau_k; \widetilde{l}_k), \tag{1}$$

where

$$\phi(s; l) = \begin{cases} 1, & \text{if } s \in [0, l),\ l \in (0, \infty) \\ 0, & \text{otherwise} \end{cases}. \tag{2}$$

Process (1) was justified in [3] as conditional linear random process (CLRP) in terms of its properties [7], and was represented in its form as a Stieltjes integral, compatible with (1):

$$\xi(t)=\int_{-\infty}^{\infty}\phi(t-\tau;\tilde{l}_{\pi(t)})\,\mathrm{d}\,\pi_\alpha(\tau),\ t\in(-\infty,\infty), \tag{3}$$

where $\phi(t-\tau;\tilde{l}_{\pi(t)})$ — random function, a kernel [7]; $\pi(t)$ — nonhomogeneous simple Poisson process with unit jumps in time moments $\tau_{\pi(t)}$, which values are used as indexes of unit water consumptions, $\pi(t)\in\mathbb{Z}$; $\pi_\alpha(t)$ — generating process: compound Poisson process, having jumps of size $\alpha_{\pi(t)}$ at the same time moments as $\pi(t)$, probability $\mathbf{P}(\pi(0)=0)=1$, variance $\mathbf{Var}(\pi_\alpha(t))<\infty$.

According to [3], finite-dimension characteristic functions of process (3):

$$f_\xi(u_1,u_2,\ldots,u_n;t_1,t_2,\ldots,t_n)=\mathbf{E}\,e^{i\sum_{k=1}^{n}u_k\xi(t_k)}=\mathbf{E}\,e^{i\sum_{k=1}^{n}u_k\int_{-\infty}^{\infty}\phi(t_k-\tau;\tilde{l}_{\pi(t_k)})\mathrm{d}\,\pi_\alpha(\tau)}, \tag{4}$$

$$\ln f_\xi(u_1,u_2,\ldots,u_n;t_1,t_2,\ldots,t_n)=\int_{-\infty}^{\infty}\int_{0}^{\infty}\int_{0}^{\infty}(e^{ix\sum_{k=1}^{n}u_k\phi(t_k-\tau;y)}-1)\,\mathrm{d}\,F_\alpha(x)\,\mathrm{d}\,F_l(y)\,\lambda(\tau)\,\mathrm{d}\,\tau,$$

are T-periodic by their time arguments:

$$f_\xi(u_1,u_2,\ldots,u_n;t_1,t_2,\ldots,t_n)=f_\xi(u_1,u_2,\ldots,u_n;t_1+T,t_2+T,\ldots,t_n+T),n\in\mathbb{N}, \tag{5}$$

since intensity $\lambda(\tau)$ of unit water consumers connection to a water provision system, is (theoretically) daily periodic: $\lambda(\tau)=\lambda(\tau+T)$, $T=24\,\text{h}$. Thus, process (3) is cyclostationary [7].

In [3], hourly water consumption ξ_t, $t\in\mathbb{Z}$ was represented as periodic autoregressive (PAR) time series (with period $T=24\,\text{h}$), adequate to (3) in terms of linear forecasting approach [8]:

$$\xi_t=\int_{t-1}^{t}\xi(s)ds,\ t\in\mathbb{Z},\quad \xi_t=m_t+\bar{\xi}_t,\quad \bar{\xi}_t=\sum_{k=1}^{p}a_{k,t}\,\bar{\xi}_{t-k}+b_t\,\eta_t, \tag{6}$$

where m_t — periodic mean of hourly water consumption: periodic (with period T) time series, expectation $\mathbf{E}\,\xi_{t+kT}=m_t$, $t\in[1,T]$, $k\in\mathbb{Z}$ and $m_t=m_{t+T}\,\forall t$; $\bar{\xi}_t$ — centered PAR time series with period T, $\mathbf{E}\,\bar{\xi}_t=0,\forall t$; $a_{k,t}$ — a-parameters of PAR: weights of dependency from sample $\{\bar{\xi}_k|k\in[t-p,t-1]\}$, containing previous p number of values before $\bar{\xi}_t$, $a_{k,t}=a_{k,t+nT}\,\forall t$, $k\in[1,p]$, $n\in\mathbb{Z}$; p — PAR model order: number of a-parameters, $p\in\mathbb{N}$; b_t — b-parameter of PAR: periodic standard deviation of generating time series (a white noise $b_t\,\eta_t$, which is a stochastic component of PAR model), $b_t=b_{t+T}\,\forall t$, $\mathbf{E}\,b_t=0$, $\mathbf{Var}(b_t)<\infty$; η_t — basic white noise, $\mathbf{E}\,\eta_t=0$, $\mathbf{Var}(\eta_t)=1$.

A method of hourly water consumption operative forecasting was proposed in [3]. According to this method, an operative forecast (one hour ahead prediction value) is a point estimate $\bar{\xi}_t$ of stochastic value ξ_t, and is determined from: recent p number of centered values $\{\bar{\xi}_k|k\in[t-p,t-1]\}$ of hourly water consumption historical data, mean estimates $\hat{m}_t$, and a-parameters estimates $\hat{a}_{k,t}$:

$$\hat{\xi}_t=\hat{m}_t+\hat{\bar{\xi}}_t,\quad \hat{\bar{\xi}}_t=\sum_{k=1}^{p}\hat{a}_{k,t}\,\bar{\xi}_{t-k}. \tag{7}$$

For each period index t, mean estimates $\hat{m}_t$ are determined from historical data as:

$$\hat{m}_t=\frac{1}{q}\sum_{k=0}^{q-1}\xi_{t+kT},t=[1,T],\hat{m}_t=\hat{m}_{t+nT},n\in\mathbb{Z}, \tag{8}$$

$$q=\left[\frac{n_\xi-t-1}{T}\right], \tag{9}$$

where q — size of historical data periodic sample for period t; n_ξ — size of all historical data; $[x]$ — floor integer of x.

The a-parameters estimates $\hat{a}_{k,t}$ are determined from centered (by periodic mean estimates) historical data $\bar{\xi}_t=\xi_t-\hat{m}_t\,\forall t$, by solving the matrix equation:

$$\hat{\mathbf{A}}_t=(\mathbf{W}_t^{\mathbf{T}}\mathbf{W}_t)^{-1}\mathbf{W}_t^{\mathbf{T}}\mathbf{X}_t,\ t=[1,T], \tag{10}$$

$$\hat{\mathbf{A}}_t^{\mathbf{T}}=(\hat{a}_{1,p+t}\quad \hat{a}_{2,p+t}\quad \ldots\quad \hat{a}_{p,p+t}),$$

$$\mathbf{X}_t^{\mathbf{T}}=(\bar{\xi}_{p+t}\quad \bar{\xi}_{p+t+T}\quad \ldots\quad \bar{\xi}_{p+t+(q-1)T}),$$

$$\mathbf{W}_t=\begin{pmatrix} \bar{\xi}_{t+p-1} & \bar{\xi}_{t+p-2} & \cdots & \bar{\xi}_t \\ \bar{\xi}_{t+T+p-1} & \bar{\xi}_{t+T+p-2} & \cdots & \bar{\xi}_{t+T} \\ \cdots & \cdots & \ddots & \cdots \\ \bar{\xi}_{t+(q-1)T+p-1} & \bar{\xi}_{t+(q-1)T+p-2} & \cdots & \bar{\xi}_{t+(q-1)T} \end{pmatrix}. \quad (11)$$

Mean estimates $\hat{m}_t$ and *a*-parameters estimates $\hat{a}_{k,t}$ are improved every hour for corresponding period index t.

The *b*-parameter estimates $\hat{b}_t$ are used once to determine the best-fit PAR order p by applying the method of minimal description length (MDL) [9], and was calculated from centered (by mean estimate) historical data and *a*-parameters estimates $\hat{a}_{k,t}$:

$$\hat{b}_{p+t}=\frac{1}{q-1}\sum_{x=0}^{q-1}\left(\bar{\xi}_{p+xT+t}-\sum_{k=1}^{n}\hat{a}_{k,p+t}\bar{\xi}_{p+xT+t-k}\right)^2, t=[1,T], \hat{b}_t=\hat{b}_{t+nT}, n\in\mathbb{Z}. \quad (12)$$

A method of hourly water consumption operative interval forecasting was proposed in [3]. According to it, each prediction interval is determined from prediction errors $\varepsilon_t, t\in\mathbb{Z}$. Each prediction error ε_t is a difference of hourly water consumption ξ_t from its most recent forecast $\hat{\xi}_t$ (which is the only one in operative forecasting):

$$\varepsilon_t=\xi_t-\hat{\xi}_t. \quad (13)$$

Analysis of (13) using known "portmanteau" test [10] has shown that its empirical distribution is not Gaussian. Therefore, the interval forecasting is based on methods of non-parametric statistics. Thus, for some preset confidence probability γ, prediction interval $[L_t, H_t]$ of water consumption ξ_t, is:

$$L_t=\hat{\xi}_t+h_{\varepsilon,\frac{1-\gamma}{2}}; \quad (14)$$
$$H_t=\hat{\xi}_t+h_{\varepsilon,\frac{1+\gamma}{2}},$$

where L_t — lower bound of prediction interval; H_t — upper bound of prediction interval; $h_{\varepsilon,x}$ — *x*-quantile estimate of time series ε_t distribution (known to be symmetric):

$$h_{\varepsilon,x}=\frac{\varepsilon_{([n_\varepsilon x+1])}-\varepsilon_{([n_\varepsilon(1-x)+1])}}{2}, \quad (15)$$

where $\varepsilon_{(k)}$ — *k*-th order statistic of time series ε_t (array of all its values sorted ascending); n_ε — size of $\varepsilon_{(k)}$; $[x]$ — floor integer of *x*.

The above hourly water consumption operative interval forecasting method was developed into information technology and tested on an urban water provision enterprise. Two versions of software, implementing the PAR parameter high-performance estimation parallel algorithm, had been developed and split-tested. Those versions differ by the following parallel computation technologies.

- OpenCL — allows more precise computations (double-precision floating point), but requires specific drivers (i.e. CUDA for nVidia) and Node.js OpenCL bindings ("node-opencl" [11] was used)
- OpenGL — utilizes vertex shaders and GLSL and is widely available without specific computational drivers. It may use WebGL interface of modern web-browsers (this is handy to offload the computations onto the web clients), and headless Node.js module [12]; yet this method is limited in precision (usually only single-precision floating point computations are available)

Despite the simplicity of OpenGL-based software integration, the authors considered it's precision as insufficient for big ranges of historical data, used for PAR parameters estimation, so this version of software has been deprecated in favor of its OpenCL-based version.

3. Proposed methodology

The improvement, described in this paper, concerns updating (8) and (10) without recalculation from scratch. This is achieved by using iterative methods of real-time estimates updating, programmed into a one-pass streaming algorithm, which is based on iterative data processing, without need to store them all (only a small recent fixed-length subset of them is stored, which is a sliding window) [13]. The OpenCL-based algorithm will be used once for initial *a*-parameter estimation (10); all subsequent updating will be done using the proposed iterative method.

3.1. Mean estimates updating

Mean estimation method, used in water consumption forecasting information technology, can be depicted as follows: for period iteration q (which is algorithm iteration $t+(q-1)T$), equation (8) is represented as:

$$\hat{m}_t=\frac{S_t}{q},\quad t\in[1,T], \tag{16}$$

where S_t — cumulative sum of data samples by period index t, defined as:

$$S_t=\sum_{k=0}^{q-1}\xi_{t+kT}. \tag{17}$$

Persistent variables t, q, and $\{S_t|t\in[1,T]\}$ are defined and preserved between algorithm iterations in a fail-safe storage (i.e. "Redis" key-value storage), and in runtime cache. Such estimation algorithm consumes only 2 integer, and T double precision floating point values of memory and storage.

3.2. The *a*-parameters estimates updating

According to [4], *a*-parameters estimates $\hat{a}_{k,t}$, $k\in[1,p]$, $t\in[1,T]$, can be improved, using least squares updating method, which can be depicted in matrix form, for period iteration q (not less than p), as:

$$\begin{gathered}\hat{\mathbf{A}}_{t,q+1}=\hat{\mathbf{A}}_{t,q}+\mathbf{P}_{t,q}\mathbf{Q}_{t,q}\left(\mathbf{Q}^{\mathbf{T}}_{t,q}\mathbf{P}_{t,q}\mathbf{Q}_{t,q}+1\right)^{-1}\left(\xi_{p+t+qT}-\mathbf{Q}^{\mathbf{T}}_{t,q}\hat{\mathbf{A}}_{t,q}\right),\\ \mathbf{P}_{t,q+1}=\mathbf{P}_{t,q}-\mathbf{P}_{t,q}\mathbf{Q}_{t,q}\left(\mathbf{Q}^{\mathbf{T}}_{t,q}\mathbf{P}_{t,q}\mathbf{Q}_{t,q}+1\right)^{-1}\mathbf{Q}^{\mathbf{T}}_{t,q}\mathbf{P}_{t,q},\\ \mathbf{Q}^{\mathbf{T}}_{t,q}=\left(\bar{\xi}_{t+qT+p-1}\quad\bar{\xi}_{t+qT+p-2}\quad\cdots\quad\bar{\xi}_{t+qT}\right),\\ q\geq p,\quad t\in[1,T],\end{gathered} \tag{18}$$

where $\hat{\mathbf{A}}_{t,q}$ — column matrix of *a*-parameters estimates (10) before an update (for iteration q of period index t); $\hat{\mathbf{A}}_{t,q+1}$ — updated column matrix of *a*-parameters estimates (for iteration $q+1$ of period index t); $\mathbf{Q}_{t,q}$ — column matrix of size $p\times 1$; $\mathbf{P}_{t,q}$ — symmetric square matrix of size $p\times p$.

In (18), $\hat{\mathbf{A}}_{t,q}$ and $\mathbf{P}_{t,q}$ are defined as recurrence relations. According to [4], their initial values can be calculated from scratch by using the historical data, using matrix equations, compatible with (10):

$$\begin{gathered}\hat{\mathbf{A}}_{t,q}=\mathbf{P}_{t,q}\mathbf{W}^{\mathbf{T}}_{t,q}\mathbf{X}_{t,q},\\ \mathbf{P}_{t,q}=\left(\mathbf{W}^{\mathbf{T}}_{t,q}\mathbf{W}_{t,q}\right)^{-1},\\ \mathbf{W}_{t,q}=\begin{pmatrix}\bar{\xi}_{t+p-1} & \bar{\xi}_{t+p-2} & \cdots & \bar{\xi}_{t}\\ \bar{\xi}_{t+T+p-1} & \bar{\xi}_{t+T+p-2} & \cdots & \bar{\xi}_{t+T}\\ \cdots & \cdots & \ddots & \cdots\\ \bar{\xi}_{t+(q-1)T+p-1} & \bar{\xi}_{t+(q-1)T+p-2} & \cdots & \bar{\xi}_{t+(q-1)T}\end{pmatrix},\\ \mathbf{X}^{\mathbf{T}}_{t,q}=\left(\bar{\xi}_{p+t}\quad\bar{\xi}_{p+t+T}\quad\cdots\quad\bar{\xi}_{p+t+(q-1)T}\right).\end{gathered} \tag{19}$$

According to (18), $\hat{\mathbf{A}}_{t,q}$ can be additively updated. So, its current state can be preserved in persistent variables $\{\hat{\mathbf{A}}_t|t\in[1,T]\}$. In each algorithm iteration, deltas

$$\Delta\hat{\mathbf{A}}_{t,q}=\mathbf{P}_{t,q}\mathbf{Q}_{t,q}\left(\mathbf{Q}^{\mathbf{T}}_{t,q}\mathbf{P}_{t,q}\mathbf{Q}_{t,q}+1\right)^{-1}\left(\xi_{p+t+qT}-\mathbf{Q}^{\mathbf{T}}_{t,q}\hat{\mathbf{A}}_{t,q}\right) \tag{20}$$

may be calculated for period index t into a temporary matrix $\Delta\hat{\mathbf{A}}$, and used to update matrix $\hat{\mathbf{A}}_t$ in-place. Variables $\{\hat{\mathbf{A}}_t|t\in[1,T]\}$ consume pT double precision floating point values of memory and storage.

The same, additive updating is also applied to $\mathbf{P}_{t,q}$, current state of which can be preserved in persistent variables $\{\mathbf{P}_t | t \in [1, T]\}$. In each algorithm iteration, deltas

$$\Delta \mathbf{P}_{t,q} = -\mathbf{P}_{t,q} \mathbf{Q}_{t,q} \left(\mathbf{Q}_{t,q}^{\mathrm{T}} \mathbf{P}_{t,q} \mathbf{Q}_{t,q} + 1 \right)^{-1} \mathbf{Q}_{t,q}^{\mathrm{T}} \mathbf{P}_{t,q} \quad (21)$$

may be calculated for period index t into a temporary matrix $\Delta \mathbf{P}$, and used to update matrix $\mathbf{P}_t$ in-place. Variables $\{\mathbf{P}_t | t \in [1, T]\}$ consume $p^2 T$ double precision floating point values of memory and storage. Also, they can be compressed by deduplicating the symmetric entries of matrices, to consume $\frac{T\, p(1+p)}{2}$ double precision floating point values.

As it concluded from [4], the current state of matrices $\mathbf{Q}_{t,q}$ can be represented by a single sliding window [13]: on each algorithm iteration, one element is appended to its rear (this value is the current hourly water consumption value ξ), and one element is removed from its front (the oldest value in it). An iterable queue of fixed length p is enough to preserve the state of this sliding window between algorithm iterations. This queue can be defined in persistent variable $\mathbf{Q}$ (which is considered in calculations as column matrix, where the bottom entry of the matrix is the front element of the queue). It consumes p double precision floating point values of memory and storage.

Persistent variables t, $\mathbf{Q}$, $\{\mathbf{P}_t | t \in [1, T]\}$, and $\{\hat{\mathbf{A}}_t | t \in [1, T]\}$ are defined and preserved between algorithm iterations in a fail-safe storage, and in runtime cache. Such estimation algorithm consumes 1 integer and $T(p^2 + p + 1)$ (or $T(\frac{p(3+p)}{2} + 1)$ with deduplication of symmetric entries) double precision floating point values of memory and storage.

3.3. Streaming algorithm of operative forecasting method

The *a*-parameters estimation algorithm uses mean estimates, and the operative forecast from previous iteration. So, streaming algorithms for mean and *a*-parameters estimation are combined along with an operative forecasting algorithm. The generalized algorithm is optimized by computation complexity to minimize the number of floating-point divisions and multiplications, thus conserving the cost of CPU usage in serverless computing environments, for which the algorithm is targeted. It's a one-pass streaming algorithm, which means that it inputs each hourly water consumption value only once. This value is latched to a sliding window $\mathbf{Q}$, and persists there only during next p iterations. Thus, algorithm iterations of the improved forecasting method don't require the whole array of historical data values anymore.

Streaming algorithm of water consumption operative interval forecasting has two phases [14]: setup and iteration. Setup phase is performed at start; then, after its successful completion, algorithm iterations (recurring runs of iteration phase) are performed. The algorithm doesn't need a wrap-up phase, as persistent variables are stored after each iteration to a fail-safe storage. Algorithm can be applied to any water consumption user, for which the scope of available contiguous historical data $\xi_t, T \in [1, n_\xi]$ allows inversion in matrix $\mathbf{P}_{t,q}$ calculation (19), i.e. matrix $\left(\mathbf{W}_{t,q}^{\mathrm{T}} \mathbf{W}_{t,q}\right)$ is not singular.

Subsequent iteration phase runs for the same user must not overlap or fail: each next run must happen only after successful completion of its predecessor. So these runs must not be triggered directly by hourly water consumption value arrival, but backed by fail-safe execution queue (i.e. "Google PubSub"); so that arrival event pushes the input value to this queue and then the queued values get processed strictly sequentially by the platform. If the run fails, it should be retried until the success. Such architecture may lead to the chokes, especially when backing services experiencing long-term connectivity issues. To minimize such risks, all backing services of information technology (i.e. fail-safe storage) must persist on the same platform, region, and network; and forecast demanding client should either be backed by outgoing queue, or do not require contiguous forecast delivery (treating delivery failures as success, allowing algorithm to continue).

This algorithm may be horizontally scaled by water consumption users, as their water consumptions (and thus, parameters and forecasts) are independent.

3.3.1. Setup phase

This phase must be performed prior to recurring runs of the iteration phase for each user. It organizes the operating state of computing environment, and prepares the persistent variables by the following actions.

- Reserving space for them in fail-safe storage
- Reserving space for them in runtime cache (if setup and iterative phase runtimes are sharing operative memory within one execution space)
- Setting their initial values
- Saving them to runtime cache and fail-safe storage
- Using them for initial estimation and forecasting (optional)

Setup phase steps of water consumption streaming algorithm are as follows.

1. Input initial historical data array $\xi_t, T \in [1, n_\xi]$
2. Define persistent variables t, q, $\{S_t | t \in [1, T]\}$, $\{\hat{m}_t | t \in [1, T]\}$, $\mathbf{Q}$, $\{\mathbf{P}_t | t \in [1, T]\}$, $\{\hat{\mathbf{A}}_t | t \in [1, T]\}$, and $\hat{\xi}$; and reserve them in storage
3. Calculate period index $t := (n_\xi - 1) \bmod T + 1$
4. Calculate period iteration index $q := \left[\dfrac{n_\xi - t - 1}{T} \right]$
5. Calculate cumulative sums:
   a. $\left\{ S_u = \sum_{k=0}^{q-1} \xi_{t+kT} | u \in [1, t] \right\}$,
   b. $\left\{ S_u = \sum_{k=0}^{q-2} \xi_{t+kT} | u \in [t+1, T] \right\}$
6. Calculate mean estimates:
   a. $\left\{ \hat{m}_k = \dfrac{S_k}{q} | k \in [1, t] \right\}$,
   b. $\left\{ \hat{m}_k = \dfrac{S_k}{q-1} | k \in [t+1, T] \right\}$
7. Define matrices $\{\mathbf{W}_t | t \in [1, T]\}$ and column matrices $\{\mathbf{X}_t | t \in [1, T]\}$ as views of historical data array $\xi_t, t \in \mathbb{Z}$, using functions:
   $\mathrm{W}_t(y, x) = \overline{\xi}_{t+p-x+(y-1)T} = \xi_{t+p-x+(y-1)T} - \hat{m}_{t+p-x}$,
   $\mathrm{X}_t(y) = \overline{\xi}_{p+t+(y-1)T} = \xi_{p+t+(y-1)T} - \hat{m}_{t+p-x}$,
   where $\mathrm{W}_t(y, x)$ — row *y* column *x* entry of matrix $\mathbf{W}_t$; $\mathrm{X}_t(y)$ — row *y* entry of $\mathbf{X}_t$
8. Define matrices $\{\mathbf{W}_t^{\mathbf{T}} | t \in [1, T]\}$ as views of matrices $\{\mathbf{W}_t | t \in [1, T]\}$, using function $\mathrm{V}_t(y, x) = \mathrm{W}_t(x, y)$,
   where $\mathrm{V}_t(y, x)$ — row *y* column *x* entry of $\mathbf{W}_t^{\mathbf{T}}$
9. Define variables $\{\mathbf{S}_k | k \in [1, T]\}$, and calculate them, using a parallel computing algorithm for matrix multiplication [12], as:
   a. $\{\mathbf{S}_k = \mathbf{W}_{k,q}^{\mathbf{T}} \mathbf{W}_{k,q} | k \in [1, t]\}$,
   b. $\{\mathbf{S}_k = \mathbf{W}_{k,q-1}^{\mathbf{T}} \mathbf{W}_{k,q-1} | k \in [t+1, T]\}$,
   where $\mathbf{W}_{k,n} = \mathrm{W}_k(y, x)|_{x \in [1, p], y \in [1, n]}$; $\mathbf{W}_{k,n}^{\mathbf{T}} = \mathrm{V}_k(y, x)|_{x \in [1, p], y \in [1, n]}$
10. Calculate matrices $\{\mathbf{P}_k = \mathbf{S}_k^{-1} | k \in [1, T]\}$, using a parallel computing algorithm for matrix inversion [12].
11. Create queue $\mathbf{Q}$, and fill it with elements $\{\overline{\xi}_{t+qT+k} = \xi_{t+qT+k} - \hat{m}_{t+k} | k \in [0, p-1]\}$, inserted in order of growing k
12. Calculate *a*-parameters estimates:
   a. $\{\hat{\mathbf{A}}_k = \mathbf{P}_k \mathbf{W}_{k,q}^{\mathbf{T}} \mathbf{X}_{k,q} | k \in [1, t]\}$,
   b. $\{\hat{\mathbf{A}}_k = \mathbf{P}_k \mathbf{W}_{k,q-1}^{\mathbf{T}} \mathbf{X}_{k,q-1} | k \in [t+1, T]\}$,

where $\mathbf{X}_{k,m}=\mathrm{X}_k(y)|_{y\in[1,m]}$

13. Calculate operative forecast $\hat{\xi}=\hat{m}_t+\mathbf{Q}^{\mathrm{T}}\hat{\mathbf{A}}_t$. Output it to the client

14. Receive feedback from client, that output was either successfully consumed or failed. This step is optional

15. Store modified persistent variables t, q, $\{S_t|t\in[1,T]\}$, $\{\hat{m}_t|t\in[1,T]\}$, $\mathbf{Q}$, $\{\mathbf{P}_t|t\in[1,T]\}$, $\{\hat{\mathbf{A}}_t|t\in[1,T]\}$, and $\hat{\xi}$ into runtime cache and fail-safe storage

16. Allow iterator phase runs

Variables S_u, $\hat{m}_k$, $\mathbf{S}_k$, and $\hat{\mathbf{A}}_k$ are calculated in two passes, because they can have different q within period index intervals $[1,t]$ and $[t+1,T]$, for historical data not aligned by period T. If historical data are aligned by period T, steps 5.b, 6.b, 9.b, and 12.b are not performed.

Step 14 is optional, and allows the algorithm to continue after a failure. This is so, because the failure to deliver this initial forecast won't violate contiguity of subsequent forecasts (which may be required by the client).

There is no concern about computation complexity of setup phase, as this phase is performed once per algorithm run (thus once for each water consumption user), and affords more budget for computation.

3.3.2. Iteration phase

After acquiring each next hourly water consumption value ξ from the sensor, the iteration phase is performed. Persistent variables get loaded by either of the following ways.

- From fail-safe storage — for first run on new execution context (and then are preserved in runtime cache)
- From runtime cache — for recurring runs on the same execution context

As the iteration phase is recurring in real-time, it's memory consumption and computing complexity are minimized as much as possible.

Iteration phase steps of water consumption streaming algorithm are as follows:

1. Input hourly water consumption value ξ
2. Load persistent variables t, q, $\{S_t|t\in[1,T]\}$, $\{\hat{m}_t|t\in[1,T]\}$, $\mathbf{Q}$, $\{\mathbf{P}_t|t\in[1,T]\}$, $\{\hat{\mathbf{A}}_t|t\in[1,T]\}$, and $\hat{\xi}$
3. If $t=T$: reset period index $t:=1$, increase period iteration index $q:=q+1$; else: increase period index $t:=t+1$
4. Update cumulative sum for period index t: $S_t:=S_t+\xi$
5. Update mean estimate for period index t: $\hat{m}_t:=\frac{S_t}{q}$. This needs one floating-point division
6. Define and calculate variable $\mathbf{R}=\mathbf{P}_t\mathbf{Q}$. Multiplication of these matrices may be done using the classic algorithm. It needs p^2 multiplications of floating-point values
7. Define and calculate variable $\mathbf{Y}=\mathbf{R}\left(\mathbf{Q}^{\mathrm{T}}\mathbf{R}+1\right)^{-1}$. Calculation of this expression needs $2p$ multiplications and one division (to calculate reciprocate number) of floating-point values
8. Define and calculate variable $\varepsilon=\xi-\hat{\xi}_k$. Its value is operative forecasting error
9. Define and calculate variable $\Delta\hat{\mathbf{A}}=\mathbf{Y}\varepsilon$. This needs p floating-point multiplications
10. Update a-parameters estimate $\hat{\mathbf{A}}_t:=\hat{\mathbf{A}}_t+\Delta\hat{\mathbf{A}}$
11. Define and calculate variable $\Delta\mathbf{P}=-\mathbf{Y}\mathbf{R}^{\mathrm{T}}$. This needs p^2 floating-point multiplications
12. Update matrix $\mathbf{P}_t:=\mathbf{P}_t+\Delta\mathbf{P}$
13. Push ξ to rear of queue $\mathbf{Q}$, and remove one element from its front
14. Calculate operative forecast $\hat{\xi}=\hat{m}_t+\mathbf{Q}^{\mathrm{T}}\hat{\mathbf{A}}_t$. This needs p floating-point multiplications. Output it to the client
15. Receive acknowledgment (feedback of success) from client, that output was successfully consumed. If it has failed, terminate the algorithm run
16. Store modified persistent variables t, q, $\{S_t|t\in[1,T]\}$, $\{\hat{m}_t|t\in[1,T]\}$, $\mathbf{Q}$, $\{\mathbf{P}_t|t\in[1,T]\}$, $\{\hat{\mathbf{A}}_t|t\in[1,T]\}$, and $\hat{\xi}$ to runtime cache and fail-safe storage

Computation of iteration phase needs $2p^2+4p$ multiplications and two divisions of floating-point values.

Step 15 preserves the client's concern about contiguity of subsequent forecasts. If the client doesn't have such a concern, this step does not terminate the algorithm run on forecast delivery failure.

Operative forecasting error ε (calculated on step 8) will be useful in further development of this algorithm to support interval forecasting.

3.4. Algorithm implementation and deployment

In ECMAScript (JavaScript), which is used as a high-level programming language in water consumption forecasting information technology, the streaming algorithms can be effectively implemented using asynchronous iterators [15]. The information technology is proposed to be deployed using Cloud Functions on Google Cloud Platform [16], selected as best-fit serverless computing environment. It also has a rich variety of backing services, which may be used as in-platform fail-safe storage ("Datastore", "Firestore" etc.).

4. Conclusion

Since memory consumption of iteration phase is constantly low, computing operations are simple (not so many multiplications and couple of divisions), and computing complexity is low, the algorithm iteration phase of information technology is suitable to be deployed on wide range of computing environments: dedicated servers, serverless clouds, and even embedded devices. For instance, in the studied water provision enterprise [3], this phase can be integrated directly into Fargo Maestro 100 GSM modems, which are used to transfer the hourly water consumption data.

Setup phase (even with any number of subsequent iterations) may be done externally, and then initial values of persistent variables may be copied into fail-safe memory of embedded device before running iteration phase on it. This may be used for off-line automated emergency detection, and for sealing the potentially damaged nodes of the water provision system.

Authors plan to improve this algorithm to support water consumption operative interval forecasting. This will require implementing a non-parametric quantile estimation method into both phases.

5. References

[1] A. Candelieri, D. Soldi and F. Archetti, Short-term forecasting of hourly water consumption by using automatic metering readers data, Procedia Engineering 119 (2015) 844–853. doi:10.1016/j.proeng.2015.08.948.

[2] E. Pacchin, S. Alvisi and M. Franchini, A Short-Term Water Demand Forecasting Model Using a Moving Window on Previously Observed Data, Water 9 (2017) 1–15. doi:10.3390/w9030172.

[3] T. Mykhailovych, M. Fryz, Model and Information Technology for Hourly Water Consumption Interval Forecasting, in: Proceedings of the 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2020, IEEE, Lviv-Slavske, Ukraine, 2020, pp. 341–346. doi:10.1109/TCSET49122.2020.235452.

[4] A. Kapustinskas, A. Nemura, Identification of Linear Random Processes, Vilnius „Mokslas" Publishers, 1983.

[5] R. Benítez, C. Ortiz-Caraballo, J. C. Preciado, J. M. Conejero, F. S. Figueroa, Alvaro Rubio-Largo, A short-term data based water consumption prediction approach, Energies 2019, Vol. 12, 2359 (2019), doi:10.3390/en12122359.

[6] E. W. Weisstein, Boxcar Function. From MathWorld — A Wolfram Web Resource, 2013. URL: https://mathworld.wolfram.com/BoxcarFunction.html .

[7] M. Fryz, B. Mlynko, Properties of Stationarity and Cyclostationarity of Conditional Linear Random Processes, in: Proceedings of the 15th International Conference on Advanced Trends in

Radioelectronics, Telecommunications and Computer Engineering, TCSET 2020, IEEE, Lviv-Slavske, Ukraine, 2020, pp. 166–171. doi:10.1109/TCSET49122.2020.235415.
[8] W. A. Gardner, A. Napolitano, and L. Paura, Cyclostationarity: Half a century of research, ISSN 0165-1684: Signal Processing, Elsevier, vol.86, no. 4, 639–697, 2006.
[9] S. L. Marple, Digital Spectral Analysis with Applications, New Jersey: Prentice Hall, 1987.
[10] A. I. McLeod, Diagnostic checking periodic autoregression models with application, ISSN 0143-9782: Journal of Time Series Analysis, vol. 15, no. 2, 221–233, 1995.
[11] Mikeseven, Node OpenCL, 2020. URL: https://github.com/mikeseven/node-opencl .
[12] T. Mykhailovych, GPU-EZ: Easy GLSL parallel computing on GPU.JS, 2019. URL: https://github.com/tarquas/gpu-ez .
[13] N. Alon, Y. Matias, M. Szegedy, The space complexity of approximating the frequency moments, Journal of Computer and System Sciences, 58 (1999) 137–147. doi:10.1006/jcss.1997.1545.
[14] M. Wolf, High-Performance Embedded Computing, ISSN 978-0-12-410511-9: Morgan Kaufmann, Elsevier, Second Edition, 139–200, 2014.
[15] I. Kantor, Async iteration and generators, 2021. URL: https://javascript.info/async-iterators-generators .
[16] Google Inc., Cloud Functions, 2021. URL: https://cloud.google.com/functions .