Journal of Theoretical and Applied Information Technology
30th April 2019. Vol.97. No 8

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

EFFICIENT APPROACH FOR DETERMINISTIC DATA EXTRAPOLATION FROM A CLEAN PERIODIC FUNCTION WITH PERIODIC COMPONENTS REPRESENTATION BY A SYSTEM OF LINEAR EQUATIONS

[1]RULLY SOELAIMAN, [2]STEVEN CANDRA, [3]M.M. IRFAN SUBAKTI

[1,2,3]Department of Informatics, Faculty of Information and Communication Technology, Institut Teknologi Sepuluh Nopember, Surabaya, Jawa Timur, Indonesia

E-mail: [1]rully@if.its.ac.id, [2]steven.candra5@gmail.com, [3]irfan@if.its.ac.id

ABSTRACT

Object forecasting has been a tedious task to be solved, such as money currency, stocks, and solar cycle predictions which are proved to be epitomes from objects that can be forecasted from periodic functions' characteristic. The comparison between an unoptimized approach and an optimized approach to extrapolate a clean periodic function formed from a sum of periodic functions with integral periods has been proposed. Initially, both approaches will be utilized system of linear equations to identify periodic components which will be extracted using arithmetic means from matrix multiplication. The resulting optimized approach will have fewer runtimes, less memory allocation, and larger scope of periods than the unoptimized one. Furthermore, the optimized approach with different implementation will also be discussed to show how the computational technique can impact the efficiency of the solution. Two testing models are involved in this paper: the correctness test by source-code submission to *Sphere Online Judge*, and the performance test by generating their chart of runtimes and standard deviation. These models have shown that the efficient implementation with optimized approach can be entitled as the first rank solution in *Sphere Online Judge.*

Keywords: *Matrix-Vector Multiplication, Data Extrapolation, Arithmetic Mean, Periodic Function, Central-Value*

1. INTRODUCTION

One of the most essential definitions yet unconsciously known by most people is that most of the objects today involves a lot of periodic activities; for instance, sound waves consist of frequency, velocity, and amplitude are periodic entities, blood pressure is also a periodic entity because when blood flows through arteries, its pattern is similar to a sine or cosine wave, and tidal waves created by gravitational pull between moon and sun are also periodic entities (Robbins, 2014).

The fact that periodic function today is not always formed by a single periodic function but by a sum of many periodic functions with noises makes the problem of periodic functions more interesting to be solved. Usually, this problem can be applied to money currency prediction or tsunami forecasting. In other words, a more striking summary to the problem is defined as follows: given periodic data of some elements which are generated a sum of periodic functions, predict the next data. However, the limit given to this problem is that the maximum number of periodic functions that forms this periodic function is defined and the state of the data is clean, which means that there are no noises involved in the generation of datasets. This paper focuses on how periodic data can be extrapolated when the periodic data is not fully-periodic.

A data extrapolation of processing a clean periodic data proposed in this paper involves a process of matrix-vector multiplication to normalize the data and arithmetic means to extract the feature of the periodic data. Firstly, the periodic data is normalized by obtaining arithmetic means from specific data with similar periodic components and subtracting those data with the arithmetic means calculated previously. This process is executed until all periodic components are all normalized and repeated for several times until the periodic data is convergent to zero.

Apart from providing algorithms to solve deterministic data extrapolation problem, this paper also provides a performance comparison between the unoptimized approach and the optimized approach for each method. This part will be explained in detail within Section 5 of this paper.

2. RELATED DEFINTIONS

2.1 Gaussian White Noise and Additive White Gaussian Noise(AWGN)

While white noise is defined to be a random signal having equal intensity in different frequencies, Gaussian noise is defined to be a type of noise whose probability density function equal to that of the normal distribution. Hence, Gaussian white noise is a stationary and ergodic random process with zero mean and independence between one and another. [9].

Since Gaussian white noise has the best approximation due to its independence, the model of Gaussian White Noise is often called the Additive Gaussian White Noise (AWGN). Additive Gaussian White Noise is defined to be a basic noise model used to mimic the effect of many random processes occurring in nature which can be defined using the equation as follows:

$$Y_i = X_i + Z_i \tag{1}$$

The channel of AWGN is represented by the output Y_i obtained from adding the original input of X_i to the noise Z_i. Since Z_i is Gaussian noise, Z_i is independent and identically distributed with zero mean value and variance of N. In other words, AWGN can be formulated as follows.

$$Z_i \sim N(0, \sigma^2) \tag{2}$$
$$Y_i = X_i + Z_i \tag{3}$$

2.2 Periodic Function

To represent a smooth model of the periodic functions as natural as possible, an additional basic noise model (white noise) to the function is necessary to mimic the random processes occurred in nature, which leads to another definition of a periodic function from section 2.1 as follows:

$$Z_i \sim N(0, \sigma^2) \tag{4}$$
$$f(z + \Omega) = f(z) + Z_i \tag{5}$$

A clean periodic function is defined to be an analytical function *f(z)* in which coexists a non-zero value of Ω such that for every *z* of the domain of regularity *f(z)*, *z*+Ω also belongs to the domain, and Z_i will always be zero since there are no external noises involved. In other words, a clean periodic function, which can be formulated in the following equation.

$$f(z + \Omega) = f(z) \tag{6}$$

Equation (6) is defined to be a periodic function with AWGN with zero means, one variance and zero value of Z_i [1].

2.2.1 Modulus Periodic Function

Modulus periodic function [6] is defined to be a type of periodic function that is always found in arithmetic modulus operation. Modulus periodic function can be defined using the following formula:

$$f(x) = f(x\%T) \tag{7}$$

2.3 Arithmetic Mean

Arithmetic mean is defined to be the sum of measurement divided by a total number of measurements [3]. Mathematically speaking, such a definition can be formulated as:

$$\bar{y} = \frac{\Sigma y_i}{n} \tag{8}$$

Other major characteristics of arithmetic means are also mentioned in [3] as follows.

1. It is an arithmetic average of measurements in the data set.
2. There is only one mean of a data set.
3. Its value is influenced by extreme measurements; trimming can help to reduce the degree of influence.
4. Means of subsets can be combined to determine the mean of the complete data set.
5. It is applicable to quantitative data only.

2.4 Central Limit Theorem in Arithmetic Mean

Arithmetic mean is often used to measure out the central value of a set of measurements, but it is subject to distortion due to the presence of one or more extreme values [3]. Such a notion can be proved as in the following.

$$\begin{aligned}
\bar{y} &= \frac{\Sigma y_i}{n} \\
\bar{y}n &= \Sigma y_i \\
0 &= \Sigma y_i - \bar{y}n \\
0 &= (y_1 + y_2 + \cdots + y_n) - \bar{y}n \\
0 &= (y_1 - \bar{y}) + (y_2 - \bar{y}) + \cdots + (y_n - \bar{y})
\end{aligned} \tag{9}$$

The proofing above showed that the mean of a dataset can represent a central value or a general value because the sum of difference distance between the dataset and the mean values leads to a zero value. The notion leads to the insight that the arithmetic mean of the dataset can represent the value of a whole dataset.

2.5 Matrix-Vector Multiplication

Matrix-vector multiplication is a simplified form of matrix-matrix multiplication. [4] defines matrix-matrix multiplication as in the following. If *A* is *m x r* matrix and *B* is *r x n* matrix, then the product

AB is the *m x n* matrix whose entries are determined as follows: to find the entry of row *i* and column *j* of *AB*, single out row *i* from matrix *A* and column *j* from matrix *B*. Multiply the corresponding entries from the row and column together and then add up the resulting products.

2.6 Systems of Linear Equations

A linear equation is defined to be an equation which involves variables with the power of 1 and can be formulated as follows.

$$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n = y_n \qquad (10)$$

Where a_1, a_2, a_3, a_n, and y_n are all constants and not all of them are zeros. A system of linear equations is a set of multiple linear equation with the same variables [4].

3. DESIGNATED PROBLEM

The problem, generally, is designed for problem-solvers to create a perfect extrapolation from a limited amount of data set. However, stereotype models for forecasting or prediction are always pointed out to the machine-learning approach.

The problem [5] is best described using this following statement.

Consider f(x) is a clean modulus periodic function whose formed from a sum of one or more clean modulus periodic functions whose period is at most N. The period of each f(x)'s components will always be integral (or integers) and the quotient of period of f(x)'s components can be rational. Given an N^2*-sized data set f(x), predict the other data.*

For example, given a dataset with N=3 with starting-zero index in the domain (*x*) as follows.

Table 1: An Instance Case for Input Data With N=3

x	0	1	2	3	4	5	6	7	8
f(x)	15	3	17	2	16	4	15	3	17

Since computer should have known the other values other than *x*=[0,9) from the datasets given in Table 1, problem-solvers should be able to formulate the pattern from the datasets given in Table 1 to be understandable by both human and computer.

A naïve approach to solve the problem utilizes only the pattern given in the dataset. The pattern observed from Table 1 is defined as a repeating data for every 6 data (*T*=6 or a period of 6). The pattern can be formulated using the following formula.

$$f(x) = f(x\%6) \qquad (11)$$

Equation (11) uses the constraint in which the value of *f(0)* and *f(5)* has already been defined. However, such an approach has its own weakness which will be described in subsection 3.1.

3.1. Addition of Periodic Functions with Integral Periods

One distinct characteristic of a periodic function is that if we try to sum one or more periodic functions, the results will often be periodic functions. Note that such a proposition is applied where two periods of periodic functions have both common multiple. Since the problem imposed in this paper only involves integer period of function and does not involve any radian values or π values, *h(x)* is a periodic function despite the exceptions.

Based on the problem [5] and [6], Table 1 can be represented in many ways as the sum of periodic functions from *T*=1 until *T*=3. Different representations can be viewed in Table 2 and Table 3 where $g(x)_{T=i}$ denotes the components for the sum of periodic functions whose period (*T*) is *i.*

Table 2: One Representation for Periodic Function from Table 1

x	$g(x)_{T=1}$	$g(x)_{T=2}$	$g(x)_{T=3}$	$f(x)=\sum_{i=1}^{3} g(x)_{T=i}$
0	10	5	0	15
1	10	-8	1	3
2	10	5	2	17
3	10	-8	0	2
4	10	5	1	16
5	10	-8	2	4
6	10	5	0	15
7	10	-8	1	3
8	10	5	2	17

Table 3: Other Representation for Periodic Function from Table 1

x	$g(x)_{T=1}$	$g(x)_{T=2}$	$g(x)_{T=3}$	$f(x)=\sum_{i=1}^{3} g(x)_{T=i}$
0	0	15	0	15
1	0	2	1	3
2	0	15	2	17
3	0	2	0	2
4	0	15	1	16
5	0	2	2	4
6	0	15	0	15
7	0	2	1	3
8	0	15	2	17

From the example of representations in both Table 2 and Table 3 above, it can be shown that the period of *f(x)* is the least common multiplier of each $g(x)_{T=i}$ where the value of *i is at 3 at maximum*. Because *g(x)* has a period of 3 and *g(x)* has a period of 2, *h(x)* has a period of

$$T_{f(x)} = lcm(1,2,3) = 6 \tag{12}$$

The formulation leads to the comparison between the amount of known data and the minimum amount of data for a single fully-periodic function which is known for each value as can be seen in Table 4. It also shows another notion that the number of known datasets is not enough to define fully-periodic data.

Table 4: Comparison Between the Amount of Known Data Set and Minimum Amount of Data Set Needed

N	A Known amount of data set (N^2)	Min. amount of data set for a complete periodic data set (*lcm([1,N]*)
1	1	1
2	4	2
3	9	6
4	16	12
5	25	60
6	36	60
7	49	420
8	64	840
9	81	2520
10	100	2520
11	121	27720
12	144	27720
13	169	360360
14	196	360360

According to Table 4, the minimum amount of dataset needed starting from N=5 is not enough to define a full periodic function from a sum of periodic functions whose maximum period is 5. Henceforth, the naïve approach by just observing the pattern accordingly to the data input only provides a blatant rebuttal starting N=5 because there is not enough data known from the input to be observed.

4. PROPOSED APPROACH

To extrapolate more data from a limited set of data with a limited constraint, a machine learning method to approach the solution is too far-fetched and prone to error. Because the machine learning method involves errors and random variables, such an approach will not be used, and that approach will not be discussed in this paper. On the other hand, for providing a perfect and unique prediction, a mathematical approach is used to settle out the problem. A definite approach to solve the problem is by identifying the periodic component using a system of linear equation, getting the general value from these periodic components, and then subtracting this general value to the corresponding periodic components.

4.1 General Idea

4.1.1 Identifying Periodic Components

If there exists a periodic function formed from a sum of periodic functions whose periods are at most 3 and a fragment of data set from the periodic function in Table 1, the data set can be visualized in Table 5 as follows.

Table 5: Dataset Representation

x	$g(x)_{T=1}$	$g(x)_{T=2}$	$g(x)_{T=3}$	*f(x)*= $\sum_{i=1}^{3} g(x)_{T=i}$
0	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,0]}$	15
1	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,1]}$	3
2	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,2]}$	17
3	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,0]}$	2
4	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,1]}$	16
5	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,2]}$	4
6	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,0]}$	15
7	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,1]}$	3
8	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,2]}$	17

$C_{[T,k]}$ values in Table 5 denotes the variable of unknown yet existing real values of periodic components for function $g(x)_{T=i}$ whose k value is $k = x\%i$. Furthermore, Table 6 can also be written as a system of the linear equation as follows.

$$\begin{aligned}
C_{[1,0]} + C_{[2,0]} + C_{[3,0]} &= 15 \\
C_{[1,0]} + C_{[2,1]} + C_{[3,1]} &= 3 \\
C_{[1,0]} + C_{[2,0]} + C_{[3,2]} &= 17 \\
C_{[1,0]} + C_{[2,1]} + C_{[3,0]} &= 2 \\
C_{[1,0]} + C_{[2,0]} + C_{[3,1]} &= 16 \\
C_{[1,0]} + C_{[2,1]} + C_{[3,2]} &= 4
\end{aligned} \tag{13}$$

The system of linear equations gives an insight into the existence of different periodic components on each equation. For instance, the periodic component of $C_{[3,i]}$ only exists for each domain *x* of *f(x)* that fulfils the condition of $x\%3 = i$. In other words, the set of domains for periodic component $C_{[T,k]}$ is domains (*x*) of *f(x)* that fulfils $k = x\%i$.

4.1.2 Coefficient Extraction

After identifying each set of domains of periodic components, the values from the domain sets from *f(x)* are then added and calculated for its arithmetic means. This arithmetic mean is then be subtracted for each current codomain values *f(x)* from the domain sets.

To extract the coefficients of periodic components thoroughly, the system of linear equations must fulfil the equation on both sides. Values on the left-hand side must be the same as the value on the right-hand side. Since the left-hand side's value must be the same as the right-hand side's value, the *bias* (Δ) value must also be zero. The *bias* (Δ) value is defined using the formula below:

$$\Delta = f(x) - \sum_{i=1}^{N} g(x)_{T=i} \quad (14)$$

By using the case from Table 1 as an instance, we start off by extracting the component of the biggest period of the data set. Extracting the data by using arithmetic means, the component for a periodic function, whose period is 3, can be seen through the following analysis.

$$\{15,2,15\} - \textit{period 3 component from data } 0$$
$$\{3,16,3\} - \textit{period 3 component from data } 1$$
$$\{17,4,17\} - \textit{period 3 component from data } 2$$

Each periodic component in the data set decreases the *bias* value. After obtaining its period components, the *bias* value is subtracted by arithmetic means, and the coefficient for the periodic component of period 3 is obtained from arithmetic means. For arithmetic mean's characteristic is to achieve a general value as it is described in section 2.4, arithmetic mean is used to extract the periodic component of the data set. In this case, the coefficient for each component will be as follows:

$$C_{[3,0]} = 10.66666667$$
$$C_{[3,1]} = 7.33333333$$
$$C_{[3,2]} = 12.66666667 \quad (15)$$

The identified value of $C_{[3,i]}$ decreases the *bias* (Δ) values and represents temporarily the value of a periodic whose period is 3 (Table 6).

Table 6: Dataset Representation After T=3 Extraction, First Iteration

x	*g(x)*$_{T=1}$	*g(x)*$_{T=2}$	*g(x)*$_{T=3}$	Δ
0	0	0	10.66667	4.33333
1	0	0	7.333333	-4.33333
2	0	0	12.66667	-4.33333
3	0	0	10.66667	-8.66667
4	0	0	7.333333	8.66667
5	0	0	12.66667	-8.66667
6	0	0	10.66667	4.33333
7	0	0	7.333333	-4.33333
8	0	0	12.66667	-4.33333

After the extraction of the period 3 component, the extraction of period 2 can be followed with a similar process in which impacts the new *bias* (Δ) values are generated, and new coefficients for period 2 component is obtained.

$$C_{[2,0]} = 5.2$$
$$C_{[2,1]} = -6.5 \quad (16)$$

Again, by subtracting the *bias* value sub data set by its corresponding coefficients and combined in the larger data set, the variable changes in the table can be visualized in Table 7.

Table 7: Dataset Representation After T=2 Extraction, First Iteration

x	*g(x)*$_{T=1}$	*g(x)*$_{T=2}$	*g(x)*$_{T=3}$	Δ
0	0	5.2	10.66667	-0.866667
1	0	-6.5	7.333333	2.16667
2	0	5.2	12.66667	-0.866667
3	0	-6.5	10.66667	-2.16667
4	0	5.2	7.333333	3.466667
5	0	-6.5	12.66667	-2.16667
6	0	5.2	10.66667	-0.866667
7	0	-6.5	7.333333	2.16667
8	0	5.2	12.66667	-0.866667

After the extraction for period 2 component has been done, the extraction for period 1 component is executed by a similar process and a different set of *bias*. The following tables show how the coefficient extraction is done.

Table 8: Dataset Representation

x	*g(x)*$_{T=1}$	*g(x)*$_{T=2}$	*g(x)*$_{T=3}$	Δ
0	0	0	0	15
1	0	0	0	3
2	0	0	0	17
3	0	0	0	2
4	0	0	0	16
5	0	0	0	4
6	0	0	0	15
7	0	0	0	3
8	0	0	0	17

$$C_{[3,0]} = 10.66666667$$
$$C_{[3,1]} = 7.33333333$$
$$C_{[3,2]} = 12.66666667 \quad (17)$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 9: Dataset Representation After T=3 Extraction, First Iteration

x	*g(x)T=1*	*g(x)T=2*	*g(x)T=3*	Δ
0	0	0	10.66667	4.33333
1	0	0	7.333333	-4.33333
2	0	0	12.66667	-4.33333
3	0	0	10.66667	-8.66667
4	0	0	7.333333	8.66667
5	0	0	12.66667	-8.66667
6	0	0	10.66667	4.33333
7	0	0	7.333333	-4.33333
8	0	0	12.66667	-4.33333

$$\mathrm{C}_{[2,0]} = 5.2$$
$$C_{[2,1]} = -6.5 \tag{18}$$

Table 10: Dataset Representation After T=2 Extraction, First Iteration

x	*g(x)T=1*	*g(x)T=2*	*g(x)T=3*	Δ
0	0	5.2	10.66667	-0.866667
1	0	-6.5	7.333333	2.16667
2	0	5.2	12.66667	-0.866667
3	0	-6.5	10.66667	-2.16667
4	0	5.2	7.333333	3.466667
5	0	-6.5	12.66667	-2.16667
6	0	5.2	10.66667	-0.866667
7	0	-6.5	7.333333	2.16667
8	0	5.2	12.66667	-0.866667

$$C_{[1,0]} = 8x10^{-17} \tag{19}$$

Table 11: Dataset Representation After T=1 Extraction, First Iteration

x	*g(x)T=1*	*g(x)T=2*	*g(x)T=3*	Δ
0	$8x10^{-7}$	5.2	10.66667	-0.866667
1	$8x10^{-7}$	-6.5	7.333333	2.16667
2	$8x10^{-7}$	5.2	12.66667	-0.866667
3	$8x10^{-7}$	-6.5	10.66667	-2.16667
4	$8x10^{-7}$	5.2	7.333333	3.466667
5	$8x10^{-7}$	-6.5	12.66667	-2.16667
6	$8x10^{-7}$	5.2	10.66667	-0.866667
7	$8x10^{-7}$	-6.5	7.333333	2.16667
8	$8x10^{-7}$	5.2	12.66667	-0.866667

Since $g(x)_{T=i}$ column is not nil anymore, the next iteration has the same process as the first iteration. However, the *bias* value yet still does not converge to zero value. That proposition defines the state that when $g(x)_{T=i}$'s values are added together, the sum of *g(x)* is not equal to *f(x)* yet. Hence, an iterative process of coefficient extraction is executed to reduce the *bias* value to zero.

The following tables represent what happens next to the data representations from Table 11 after *n* iterations. Only three iterations are described in the following tables.

Table 12: Dataset Representation after Second Iteration

x	*g(x)T=1*	*g(x)T=2*	*g(x)T=3*	Δ
0	$8x10^{-7}$	5.72	9.36667	-0.0866667
1	$8x10^{-7}$	-7.15	9.933333	0.216667
2	$8x10^{-7}$	5.72	11.36667	-0.0866667
3	$8x10^{-7}$	-7.15	9.36667	-0.216667
4	$8x10^{-7}$	5.72	9.933333	0.3466667
5	$8x10^{-7}$	-7.15	11.36667	-0.216667
6	$8x10^{-7}$	5.72	9.36667	-0.0866667
7	$8x10^{-7}$	-7.15	9.933333	0.216667
8	$8x10^{-7}$	5.72	11.36667	-0.0866667

Table 13: Dataset Representation after Third Iteration

x	*g(x)T=1*	*g(x)T=2*	*g(x)T=3*	Δ
0	$8x10^{-7}$	5.772	9.236667	-0.00866667
1	$8x10^{-7}$	-7.215	10.133333	0.0216667
2	$8x10^{-7}$	5.772	11.236667	-0.00866667
3	$8x10^{-7}$	-7.215	9.236667	-0.0216667
4	$8x10^{-7}$	5.772	10.133333	0.03466667
5	$8x10^{-7}$	-7.215	11.236667	-0.0216667
6	$8x10^{-7}$	5.772	9.236667	-0.00866667
7	$8x10^{-7}$	-7.215	10.133333	0.0216667
8	$8x10^{-7}$	5.772	11.236667	-0.00866667

Table 12 and Table 13 both show that after three iterations have been done, the coefficient obtained from the coefficient extraction process can be calculated as below.

a) Period 3 coefficients
$$C_{[3,0]} = 9.223666667$$
$$C_{[3,1]} = 10.1933333$$
$$C_{[3,2]} = 11.2366667$$
b) Period 2 coefficients
$$\mathrm{C}_{[2,0]} = 5.7772$$
$$\mathrm{C}_{[2,1]} = -7.215$$
c) Period 1 coefficients
$$\mathrm{C}_{[1,0]} = 8\mathrm{x}10^{-17}$$

4.1.3 Data Extrapolation

To predict the periodic function precisely, one must certain about the characteristics of a periodic function, especially what will happen if one or more periodic functions with integral periods are summed.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

By adopting from the instance of representation in Table 1, one can analyze the characteristic of a sum of periodic functions whose period is at most 3 by Table 14.

Table 14: Problem Representation With N=3

x	$g(x)_{T=1}$	$g(x)_{T=2}$	$g(x)_{T=3}$	$f(x)=\sum_{i=1}^{3} g(x)_{T=i}$
0	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,0]}$	15
1	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,1]}$	3
2	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,2]}$	17
3	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,0]}$	2
4	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,1]}$	16
5	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,2]}$	4
6	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,0]}$	15
7	$C_{[1,0]}$	$C_{[2,0]}$	$C_{[3,1]}$	3
8	$C_{[1,0]}$	$C_{[2,1]}$	$C_{[3,2]}$	17

If the coefficient values are substituted by the coefficient values obtained in section 4.1.2, Table 15's value will meet up the requirements of *f(x)* value as follows.

Table 15: Answer Proofing for Section 4.1.2

x	$g(x)_{T=1}$	$g(x)_{T=2}$	$g(x)_{T=3}$	$f(x)=\sum_{i=1}^{3} g(x)_{T=i}$
0	8×10^{-7}	5.772	9.236667	15
1	8×10^{-7}	-7.215	10.133333	3
2	8×10^{-7}	5.772	11.236667	17
3	8×10^{-7}	-7.215	9.236667	2
4	8×10^{-7}	5.772	10.133333	16
5	8×10^{-7}	-7.215	11.236667	4
6	8×10^{-7}	5.772	9.236667	15
7	8×10^{-7}	-7.215	10.133333	3
8	8×10^{-7}	5.772	11.236667	17

According to Table 15, to extrapolate x^{th} data from a limited size of data set, a definite formula can be derived and defined as follows.

$$f(x) = \sum_{T=1}^{N} C_{[T,x\%T]} \qquad (20)$$

4.2 Mathematical Approach

4.2.1 Coefficient Extraction

The process of coefficient extraction can also be represented in matrix equations; the mathematical words for the process are defined to be if A is the initial state of data set and A' is the state of data set after period N component extraction, then the equation can also be written in an equation notation as follows:

$$A' = A - (EA) \qquad (21)$$

Where E is a matrix that extracts the component of the period in the data set by arithmetic means. The equation can also be simplified as below.

$$A' = A - (E.A)$$
$$A' = (I - E).A \qquad (22)$$

Where I is an identity matrix whose size corresponds to matrix E. For instance, if the following data set of periods 3 as in Table 1 is defined as follows.

$$A = \begin{bmatrix} 15 \\ 3 \\ 17 \\ 2 \\ 16 \\ 4 \\ 15 \\ 3 \\ 17 \end{bmatrix} \qquad (23)$$

If the period 3 component needs to be extracted from the data set, the necessary thing to do is to generate E so that the component of period 3 can be extracted. The component for $C_{[3,0]}$ can be extracted by its arithmetic means using the matrix transformation below:

$$\mathrm{I} - \mathrm{E}_{C_{[3,0]}} = \begin{bmatrix} \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad (24)$$

By using the same approach on how to obtain $\mathrm{E}_{C_{[3,0]}}$, $\mathrm{E}_{C_{[3,1]}}$ and $\mathrm{E}_{C_{[3,2]}}$ can also be obtained as follows:

$$\mathrm{I} - \mathrm{E}_{C_{[3,1]}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad (25)$$

$$I - \mathrm{E}_{C_{[3,2]}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} \end{bmatrix} \quad (26)$$

Since the coefficient extraction approach affects the data independently, the matrix transformation of $(\mathrm{I} - \mathrm{E}_{C_{[3,0]}})$, $(\mathrm{I} - \mathrm{E}_{C_{[3,1]}})$, and $(\mathrm{I} - \mathrm{E}_{C_{[3,2]}})$ can be combined with matrix multiplication to be $\mathrm{E}'_{C_{[3]}}$. The result of such matrix multiplication is $\mathrm{E}'_{C_{[3]}}$ which can be shown from the matrix below:

$$\mathrm{E}'_{C_{[3]}} = \begin{bmatrix} \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} \\ -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 \\ 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 \\ 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{3} \\ -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 \\ 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & -\frac{1}{3} & 0 & 0 & -\frac{1}{3} & 0 & 0 & \frac{1}{3} \end{bmatrix} \quad (27)$$

Such a notion leads us to a definition of E matrix to aid in solving the designated problem of extraction, as below.

$$\mathrm{E}_{C_{[N]}} = \begin{bmatrix} -\frac{1}{x_0} & \cdots & -\frac{1}{x_0} & \cdots & -\frac{1}{x_0} \\ 0 & -\frac{1}{x_1} & \cdots & -\frac{1}{x_1} & \cdots \\ 0 & 0 & -\frac{1}{x_2} & \cdots & -\frac{1}{x_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -\frac{1}{x_0} & \cdots & -\frac{1}{x_0} & \cdots & -\frac{1}{x_0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad (28)$$

The variable x_n is the sum of data affected in n[th] rows. In other words, x_n is how many nonzero values in the n[th] row.

4.3 Computational Approach

4.3.1 General Idea

The main flow of the computational approach for coefficient extraction is to start off by generating matrices $E_{C_{[N]}}$ until $E_{C_{[1]}}$ for coefficient extraction from N until 1. For each matrix generated this way, extract the coefficients by using the data from the entry of the matrices. After that, multiply the matrix $\left(I - E_{C_{[i]}}\right)$ to get the new data set after extracting the period *i* component from the old data set. According to the main flow of coefficient extraction, there will be 3 functions that will be described further in section 4.5 and 4.6.

The main flow for data prediction is adopted from the formula given in section 4.1.3, as in the following.

$$f(n) = \sum_{T=1}^{N} C_{[T,n\%T]} \quad (29)$$

The main flow of coefficient extraction can also be visualized by using the flowchart in Figure 1 and Figure 2, respectively.

4.3.2 Extracting Coefficients using N^2 x N^2 matrix

Based on section 4.3.1, there are three essential functions that must be done sequentially, i.e. generate matrix function, multiplying the matrix with the data set, and extracting coefficients function.

First, the *GenerateMatrix* function is a function that generates an N^2xN^2 sized matrix as described in section 4.2.1 for each different value of T. This function is generated repeatedly, where both *bias* sets are all convergence to zero and the iteration ends at T=1. Second, the function *ExtractCoefficient* showed in Figure 1 only utilizes the matrix in *GenerateMatrix* function to obtain the arithmetic means. Finally, the function *MultiplyMatrixVector* is defined to be the function that subtracts the appointed *bias* value to the corresponding identified periodic components identified.

Journal of Theoretical and Applied Information Technology
30th April 2019. Vol.97. No 8

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

This function runs in O(N^6) runtime because it needs to fill each by each entry of the matrix. The pseudocode for this function can be seen in Figure 1.

Algorithm 1 Coefficient Extractor - N^{2}x N^2 matrix

```
procedure GenerateMatrix(T)
    for i := 0; i < N^2; i++ do
        endpoint := N^2 − (i%T) − 1
        count := (end/T) + 1
        for j := i%T; j < N^2; j+=T do
            matrix[i][j] := count
        end for
    end for
end procedure
procedure ExtractCoefficient(T)
    for i := 0; i < T; i++ do
        sum := 0
        for j := i; j < N^2; j+=T do
            sum+ = data[j]
        end for
        coefficient[T, i]+ = (sum/matrix[i][i])
    end for
end procedure
procedure MultiplyMatrixVector(T)
    for i := 0; i < N^2; i++ do
        rowAnswer := 0
        for j := i%T; j < N^2; j+=T do
            mult := −1.0/matrix[i][j]
            if i == j then mult = 1.0 + mult
            end if
            rowAnswer+ = data[j] ∗ mult
        end for
        answer.append(rowAnswer)
    end for
    Return answer
end procedure
```

Figure 1: Pseudocode for Coefficient Extraction Using N^2 x N^2 Matrix

4.3.3 Extracting Coefficients using N x N matrix

The more optimized form from section 4.3.2 can be obtained by examining the fact on how to store the arithmetic mean's denominator in the matrix. A smaller matrix size is used to reduce the heavy run time so that the runtime of matrix generation is reduced to O(N^5). This small matrix is also maximally utilized during coefficient extraction process and matrix-vector multiplication process.

$$E'_{C_{N=3}} = \begin{bmatrix} 3 & 3 & 3 \\ 5 & 4 & 0 \\ 9 & 0 & 0 \end{bmatrix} \qquad (30)$$

The difference of algorithm showed in Figure 2 and in Figure 1 is that the function *GenerateMatrix* generates an upper-triangular matrix of the positive denominator, as in the equation above (Table 1 case), once for all denominator values of *T*, and there are no other matrices generated afterwards.

Algorithm 2 Coefficient Extractor - N x N matrix

```
procedure GenerateMatrix(T)
    for i := 0; i < N^2; i++ do
        mx := T^2
        ctr := i
        for j := i%T; j < N^2; j+=T do
            matrix[i][j] = ceil(mx/ctr)
            mx− = ceil(mx/ctr)
            ctr− = 1
        end for
    end for
end procedure
procedure ExtractCoefficient(T)
    for i := 1; i < T + 1; i++ do
        sum := 0
        for j := i − 1; j < N^2; j+=T do
            sum+ = data[j]
        end for
        coefficient[T, i − 1]+ = (sum/matrix[T][i])
    end for
end procedure
procedure MultiplyMatrixVector(T)
    for i := 0; i < N^2; i++ do
        rowAnswer = 0
        for j := i%T; j < N^2; j+=T do
            mult = −1.0/matrix[T][(i%T) − 1]
            if i == j then mult = 1.0 + mult
            end if
            rowAnswer+ = data[j] ∗ mult
        end for
        answer.append(rowAnswer)
    end for
    Return answer
end procedure
```

Figure 2: Pseudocode for Coefficient Extraction Using N x N Matrix

5. TESTING

The discussed approaches on section 4 impacts on the performance of the algorithm. Two types of tests will be discussed in this section: the correctness test and the performance test.

While the performance test is done by running the source code in local computer to obtain the statistics of the program runtime which will then be compared accordingly, the correctness test will use the submission of source code in *Sphere Online Judge* as the third-party online platform for source code checker. *Sphere Online Judge* (SPOJ) serves as an online judge system that allows users to submit source code solution for any problem. This source code is, then, executed and the results from the execution are compared to the answers provided by the problem maker.

5.1 Testing Methodology

On one hand, the correctness test is performed by submitting three source codes to three different problems in *Sphere Online Judge*: *Periodic*

Journal of Theoretical and Applied Information Technology
30th April 2019. Vol.97. No 8

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Function, trip 3 (easy) [10], *Periodic Function, trip 3* [6]*, and Periodic Function, trip 5* [5]. All three problems have different constraints as below.

1. *Sphere Online Judge*: *Periodic Function, trip 3 (easy)* [10] with N <14 in 1-second runtime mean.
2. *Sphere Online Judge*: *Periodic Function, trip 3 [*6] with N < 51 with an accumulation of 11-second runtime for all test cases.
3. *Sphere Online Judge*: *Periodic Function, trip 5* [5] with N < 256 with 1-second runtime mean.

These three source codes are all different from one and another. The first implementation represents the approach in section 4.3.2 using an N^2xN^2 sized matrix to extract coefficient; furthermore, the second and the third implementation represent the approach of using an N×N sized matrix as described in 4.3.3. However, the second and the third implementation differ for the third source code in four approaches of implementation, such as:

a) Using *pass by reference* instead of *pass by value*.
b) Using *integer* variable type instead of *long long* variable type.
c) Using *return by reference* instead of *return by value*.
d) Using *static* allocation for arrays.

On the other hand, the performance test is done by first generating the input by randomizing each notable periodic component (randomizing the values of $g(x)_{T=i}$) which then added to corresponding x value producing a periodic datum for a single x. Each of these components is added creating a sequence of the periodic dataset.

This periodic dataset is then randomized and replicated to 1000 datasets for a single value of N. The observed N values ranged from N=1 to N=40, which means that there will be 14000 data inputs used for performance testing.

The performance test involves two main metrics and three source codes that are used for the correctness test. The metrics measured in this paper are mean runtime metric and standard deviation metric.

5.2 Correctness Test

Figure 3, Figure 4, and Figure 5 show how the approaches used according to section 4.3.2 are scored in *Sphere Online Judge* using the three problems stated. The first source code submission has been executed at least 15 times to ensure the solution's both validity and consistency during the correctness test. Both Figure 3 and Figure 4 show a snippet of 4 submissions using the approaches in section 4.3.2; however, Figure 5 only shows a snippet of one submission of *compilation error* verdict due to memory overflow.

2018-11-29 13:33:01	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.89
2018-11-29 13:32:55	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.89
2018-11-29 13:32:51	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.91
2018-11-29 13:32:30	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.91

Figure 3: Correctness Test Using Approach of N^2xN^2 sized matrix in Periodic Function, trip 3 (easy) [10]

2018-11-29 14:18:12	Periodic function, trip 3	**accepted** edit ideone it	8.80	119M
2018-11-29 14:18:07	Periodic function, trip 3	time limit exceeded edit ideone it	-	118M
2018-11-29 14:17:59	Periodic function, trip 3	**accepted** edit ideone it	10.95	118M
2018-11-29 14:17:53	Periodic function, trip 3	time limit exceeded edit ideone it	-	119M

Figure 4: Correctness Test Using Approach of N^2xN^2 sized matrix in Periodic Function, trip 3 [6]

2018-11-29 13:45:11	Periodic function, trip 5	compilation error edit ideone it	-	-

Figure 5: Correctness Test Using Approach of N^2xN^2 sized matrix in Periodic Function, trip 5 [5]

Despite the inconsistency of correctness by using approach 4.3.2, Figure 6-8 all show the optimized approach discussed in 4.3.3 by consistent correctness by submitting the second source code. The second source code submission has also done at least 15 times, and it shows consistency during all submissions. Informatively, all figures (Figure 6-8) display only some numbers of submissions among 15 submissions.

2018-07-22 21:06:39	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.35	16M
2018-07-22 21:06:24	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.34	16M
2018-07-22 21:06:20	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.37	16M
2018-07-22 21:06:17	Periodic function, trip 3 (easy)	**accepted** edit ideone it	0.36	16M

Figure 6: Correctness Test Using Approach of NxN sized matrix in Periodic Function, trip 3 (easy) [10]

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2018-07-22 21:23:28	Periodic function, trip 3	accepted edit ideone it	0.22	16M
2018-07-22 21:23:26	Periodic function, trip 3	accepted edit ideone it	0.21	16M
2018-07-22 21:23:22	Periodic function, trip 3	accepted edit ideone it	0.22	15M
2018-07-22 21:23:19	Periodic function, trip 3	accepted edit ideone it	0.21	16M

Figure 7: Correctness Test Using Approach of NxN sized matrix in Periodic Function, trip 3 [6]

2018-11-29 15:36:25	Periodic function, trip 5	accepted edit ideone it	11.28	23M
2018-11-29 15:36:19	Periodic function, trip 5	accepted edit ideone it	11.30	27M
2018-11-29 15:36:16	Periodic function, trip 5	accepted edit ideone it	11.42	27M
2018-11-29 15:36:10	Periodic function, trip 5	accepted edit ideone it	11.06	27M

Figure 8: Correctness Test Using Approach of NxN sized matrix in Periodic Function, trip 5 [5]

The last implementation that will be used for testing is the implementation of section 4.3.3 with the differences in the implementation techniques, which utilizes more optimized approaches of implementation, as it has been stated in the previous section. Figure 9-11 show the verdict for submissions in *Sphere Online Judge*.

2018-11-29 17:53:07	Periodic function, trip 3 (easy)	accepted edit ideone it	0.05	15M
2018-11-29 17:53:01	Periodic function, trip 3 (easy)	accepted edit ideone it	0.05	15M
2018-11-29 17:52:58	Periodic function, trip 3 (easy)	accepted edit ideone it	0.05	16M
2018-11-29 17:52:53	Periodic function, trip 3 (easy)	accepted edit ideone it	0.05	16M

Figure 9: Correctness Test Using Different Approach of NxN sized matrix in Periodic Function, trip 3 (easy) [10]

2018-12-11 05:51:48	Periodic function, trip 3	accepted edit ideone it	0.00	18M
2018-12-11 05:51:44	Periodic function, trip 3	accepted edit ideone it	0.00	18M
2018-11-29 17:49:45	Periodic function, trip 3	accepted edit ideone it	0.00	18M
2018-11-29 17:49:41	Periodic function, trip 3	accepted edit ideone it	0.00	18M

Figure 10: Correctness Test Using Different Approach of NxN sized matrix in Periodic Function, trip 3 [6]

2018-09-02 02:49:00	Periodic function, trip 5	accepted edit ideone it	0.64	18M
2018-09-02 02:46:47	Periodic function, trip 5	accepted edit ideone it	0.71	18M
2018-09-02 02:43:15	Periodic function, trip 5	accepted edit ideone it	0.63	17M
2018-09-01 19:41:14	Periodic function, trip 5	accepted edit ideone it	0.59	17M

Figure 11: Correctness Test Using Different Approach of NxN sized matrix in Periodic Function, trip 5 [5]

The memory described in the third implementation is less than those described in the first and the second implementation. Figure 9-11 give an insight that the third implementation has the least amount of memory involved, i.e. 16M. The second implementation, which scopes from Figure 6-8, allocates memory size for about 20M; hence, the second implementation allocates more memory than the third implementation.

The first implementation is not considered as a valid solution for the problem [5] since the computer allocation for memory in the array is less than the amount of memory needed, i.e. 10^6 amounts of the element in an array.

Figure 12 shows how the solution is ranked among users in *Sphere Online Judge*. It is showed that the efficient implementation (the third implementation) takes the best solution to the first rank on the submission.

RANK	USER	RESULT	TIME	MEM
1	Steven Candra	accepted	0.59	17M
2	Mateusz Radecki	accepted	0.60	94M
3	Rully Soelaiman	accepted	0.62	17M
4	Kamil Debowski	accepted	0.72	17M

Figure 12: Solution Ranking in Sphere Online Judge: Periodic Function, trip 5 [5]

5.3 Performance Test

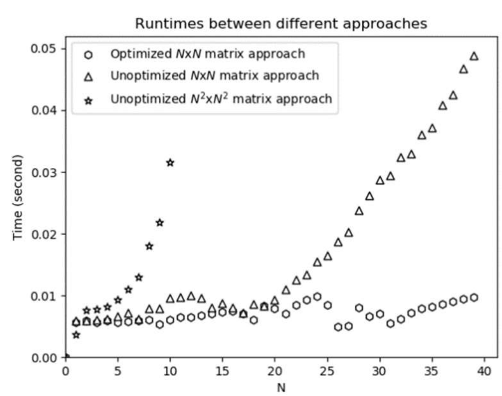

Figure 13: Running time mean of each approach for data extrapolation

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 13 shows a very distinct difference in the performance of each approach. This difference asserts the notion that from N=11, the approach in section 4.3.2 (the first implementation) has a skyrocketing runtime due to its complexity ($O(N^6)$). The performance of the other approach, which is described in section 4.3.3, climbs gradually from N=20 and proves that this approach (the second implementation) is better than the unoptimized approach that uses N^2xN^2 matrix.

Yet the third implementation uses the same approach as the second implementation does, the third implementation, however, remains stable from N=1 to N=40. By tweaking the implementation approach only, the performance of the third implementation remains much more stagnant with small increments by each N values than other approaches.

5.4 Standard Deviation on Unoptimized Approach (using N^2xN^2 matrix)

Figure 14 shows an incrementing value from each value of N, which those approved the notions that there will be a steep increase by the time N reaches more than 10.

This scatter plot also approved that by the time N gets larger, the runtime of the program may get longer due to the standard deviation value of N starting from 12 reaches 0.02 seconds. This scatter plot, furthermore, also tells that the mean of runtimes from N=10 starts to spread out by 0.15 second, and the runtimes reach an instability starting from N=20 with the difference of 0.05 second.

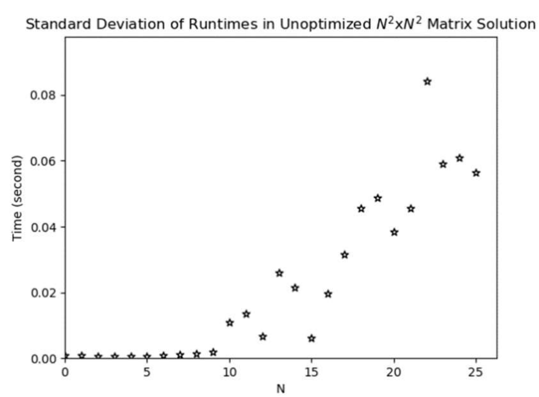

Figure 14: Standard deviation of runtimes for the approach using N^2xN^2 matrix

5.5 Performance on Optimized Approach (using NxN matrix)

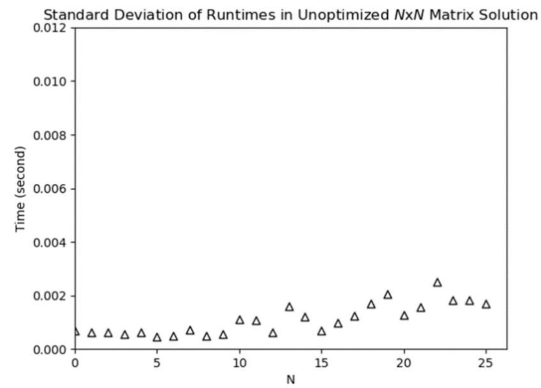

Figure 15: Standard deviation of runtimes for the N×N matrix approach in the second implementation

Figure 15-16 show the standard deviations of runtimes from the approach of using a N×N matrix as the base matrix to extract coefficients. It is visually described that both Figure 15-16 have a very similar standard deviation. This similarity in standard deviation inferred that the optimization from reducing the matrix size does give an impactful effect to the fluctuance of the runtimes of both approaches by comparing those of the first implementation.

In addition, both approaches seemingly remain stagnant at the standard deviation of 0.001 seconds. Hence, this stagnancy gives an insight that no matter how random the test cases are, the approach described in section 4.3.3 will give a spread of runtimes for about 0.001 seconds only, which is not affecting the program in a wholesome in terms of runtime.

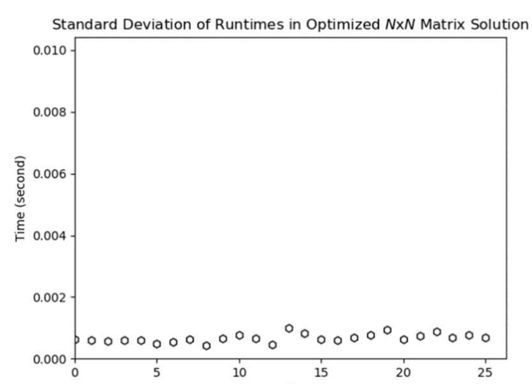

Figure 16: Standard deviation of runtimes for the NxN matrix approach in the third implementation

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

6. CONCLUSION

A new horizon of statistical approach has been shown, especially about how a system of linear equations can identify the periodic components in a periodic function and how arithmetic means can be used to obtain a representative number of populations from a sample of data. Not only does the statistical approach useful to solve the problem, but the computational approach, which describes the differences in implementation, can impact the runtime of the program significantly.

This research is far yet from over since the data provided in this case are all pure-stated data (Z_i=0) in which there are no variables which make the data volatile. There are no external factors considered in this paper since this paper is used to develop a basis for further research about data modelling and predictions.

REFERENCES:

[1] Knopp Konrad, Theory of Functions, New York, 1947.

[2] Caveny Regina, Caveny James, "Periodic Sums of Functions with Incommensurable Periods", *Mathematics Magazine*, Vol. 55, No. 5, 1982, pp. 280-285.

[3] Ott Lyman R., Longnecker Michael, An Introduction to Statistical Methods and Data Analysis, Brooks/Cole, Belmont, 2010.

[4] Anton Howard, Rorres Chris, Elementary Linear Algebra 11th Edition, Wiley, Canada, 2014.

[5] SPOJ (September 11th, 2018). PERIOD5 – Periodic Function, trip 5 [online]. Available: http://www.spoj.com/problems/PERIOD5

[6] SPOJ (September 11th, 2018). PERIOD3 – Periodic Function, trip 3 [online]. Available: http://www.spoj.com/problems/PERIOD3

[7] Pei Soo-Chang, Lu Shih-Keng, "Intrinsic Integer-Periodic Function for Discrete Periodicity Detection", *Institute of Electrical and Electronics Engineers (IEEE)*, Vol. 22, 2015, pp. 1108-1112.

[8] Cucuringu Mihai, Tyagi Hemant, "On Denoising Modulo 1 samples of function", *Proceedings of Machine Learning Research*, Vol. 84, pp. 1868 – 1876

[9] Wiley Open Library (January 22th, 2019). GWN - Gaussian White Noise [online]. Available: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780471679370.app2

[10] SPOJ (September 11th, 2018). PERIOD4 – Periodic Function, trip 3(easy) [online]. Available: http://www.spoj.com/problems/PERIOD4