Generating Synthetic Time Series Data using Java Programming

Author, Aneesh Roy

A Data Science Foundation White Paper

September 2019

---

www.datascience.foundation

Data Science Foundation
Data Science Foundation, Atlantic Business Centre, Atlantic Street, Altrincham, WA14 5NQ
Tel: 0161 926 3641 Email: admin@datascience.foundation Web: www.datascience.foundation
Registered in England and Wales 4th June 2015, Registered Number 9624670

Acknowledgement

I would like to thank Rittika Shamsuddin[Ph.D. Student, Eric Johnson School of Computer Science, University of Dallas, Texas] , Professor Balakrishnan Prabhakaran[Professor, Eric Johnson School of Computer Science, University of Dallas, Texas] and University of Dallas (UTD) for the opportunity to venture into the Data Science space. During the Summer of 2018, I was provided guidance and a lab environment to learn about Data Synthesis and MATLAB. My coding skills and Java knowledge are attributed to Summer Coding Programs offered by UTD for local schools.

Abstract

A synthetic data generation algorithm was developed that combines statistical data analysis techniques with convergence criteria to generate new synthetic data sets based upon a given time series data set. The algorithm was developed based upon prior research into data synthesis concepts and manipulation. The program was tested with a variety of time series data sets that exhibited complex distribution patterns, with the objective that the convergence is fast and should be able to run in a laptop environment. The algorithm was created to be an accessible and efficient method of synthetic data generation, for greater usability by data scientists for research. The algorithm was developed in JAVA, using standard statistical packages and methods. The workflow of the algorithm is discussed, as well as an analysis on the similarities and differences between the original data and the generated data. In addition to its single step data synthesis, also tested was how generations of synthesized data would compare if a dataset goes through the algorithm multiple times. It was found that the synthetic data algorithm had both advantages and disadvantages in its methods of numerical calculation and could possibly be manipulated further to change the similarity levels of the synthetic and original datasets. This algorithm can aid data driven research scientists with statistically generated data sets to validate their hypothesis and computational models.

Keywords: synthetic time series data; multi-modal distribution, java statistical methods

Introduction

McGraw-Hill Dictionary of Scientific and Technical Terms["Synthetic data". McGraw-Hill Dictionary of Scientific and Technical Terms. Retrieved November 29, 2009.] defines synthetic data as "any production of data applicable to a given situation that is not obtained by direct measurement." The concept of data synthesis has been present in data science for decades and has numerous applications in the industry. One of the primary applications is its ability to

provide data anonymity, where the general distribution and trends are preserved while exact measurements and identifying parameters are not. Another application of data synthesis is the ability to simulate the introduction of variability in the original data and thus expand the range of the outcome. This application of data synthesis is important in the field of medicine, where the data is not only limited but the outcome of a study is dependent on the characteristics of the patient. It is hard to capture all the characteristics of the patient, as they are limited to the exhibited symptoms that are measurable by the available devices at the time of the study. While data generation is useful in these scenarios, the algorithms require powerful hardware and thus are limited to institutions with such computational capacity. The discussed approach allows synthetic data generation in many settings due to its low processing requirements.

The algorithm was tested with a variety of time series datasets. Time series data is a collection of data that is measured in equal time intervals. For example, in the medical field, time series data is generated during a “heart stress test” with an ECG (Electrocardiogram) machine. During a stress test, the patient is required to walk (or run) and its impact on the heart rate is measured in equal intervals of time during the test. The study of time-series data is also widely applicable in other fields such as big data, logistics, advertising, etc.

As researchers push the boundaries of computational analytics to predict outcomes from measurements and trends, they are challenged by the availability of datasets to test their programs. Thus, a faster and easier data synthesis can help such researchers by generating synthesized virtual data sets which will allow them to validate their programs and thus improve the quality of their research.

This work was done under the guidance of Rittika Shamsuddin[Ph.D. Student, Eric Johnson School of Computer Science, University of Dallas, Texas] and Balakrishnan Prabhakaran[Professor, Eric Johnson School of Computer Science, University of Dallas, Texas]. This work has referenced the publication Shamsuddin et.al where the process of data synthesis was investigated using machine learning[“Virtual Patient Model: An Approach for Generating Synthetic Healthcare Time Series Data”, Shamsuddin et.al].

Data Synthesizer Models

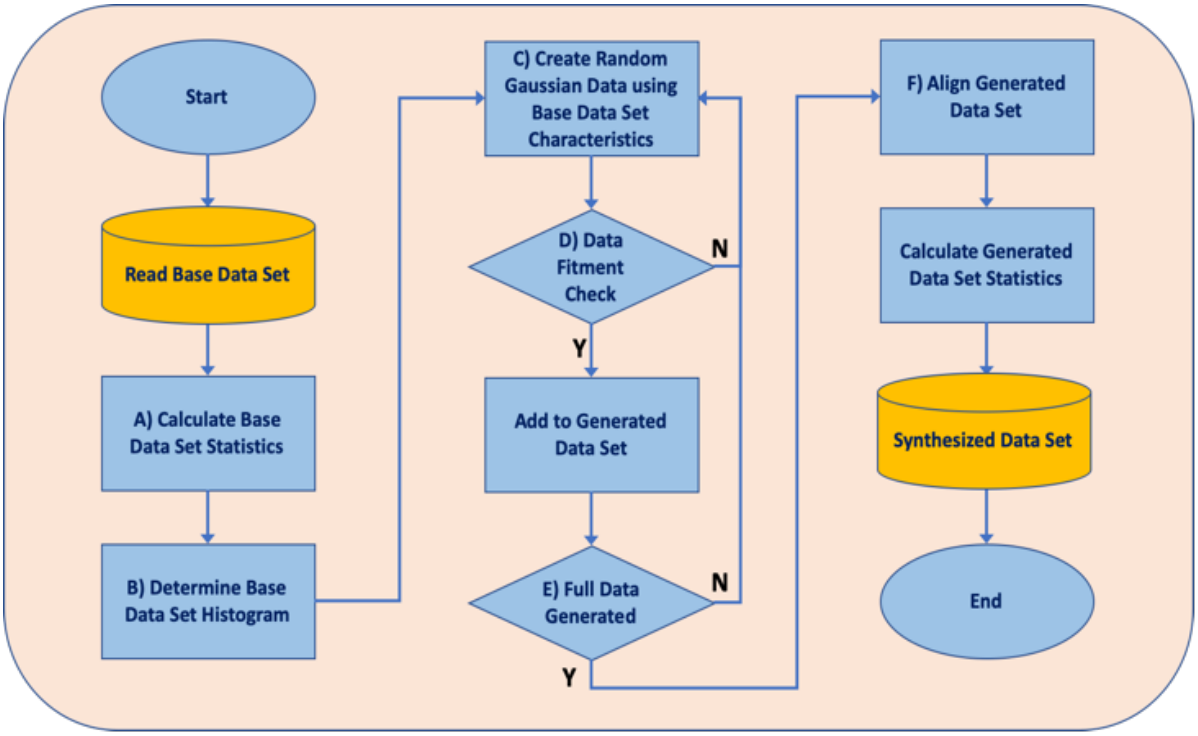
This section describes the two models developed as part of the study. The “First-Generation Data Synthesizer” generates the synthesized data from a base time-series data in a single step. The “Multi-Generation Data Synthesizer” model, described later, is an extension of the “First-Generation Data Synthesizer” where the model analyzes the pattern of the evolution of the synthesized data through many iterations.

First-Generation Data Synthesizer: The workflow for the First Gen Data Synthesizer program is shown below.

The main steps in the above workflow are elaborated below:

A. Calculate Base Data Set Statistics: The min, max, mean, standard deviation and the skewness of the original data is calculated.

B. Determine Base Data Set Histogram: A histogram is created with the base data set. The program initially calculates the bin number based upon a user defined parameter and then calculates the distribution pattern using the frequency of the data in each bin. The bin size can be varied during the execution to help the solution converge faster. For example, the initial distribution is determined based on 10 bins. If the solution does not converge in a given number of iterations, then the bin size is relaxed. For most of the calculations the synthesized data was calculated with a distribution pattern of 10 bins.



C. Create Random Gaussian Data using Base Data Set Characteristics: Here, a new data point is created using the formula shown below. Random.NextGaussian is a Java Random number generator for Gaussian distribution.
   - *New Data Value = "Random.NextGaussian()[Standard JAVA method that generates a Gaussian(normally) distributed value with mean 0.0 and standard deviation 1.0 from the random number generator's sequence]" x Standard Deviation (Base Dataset) + Mean (Base Dataset)*

D. Data Fitness Check: There are two validation steps before the data is accepted as a part of the newly synthesized dataset.
   - Check 1: The new data has to be between the determined min and max values for the generated dataset. The min and max values of the generated dataset were

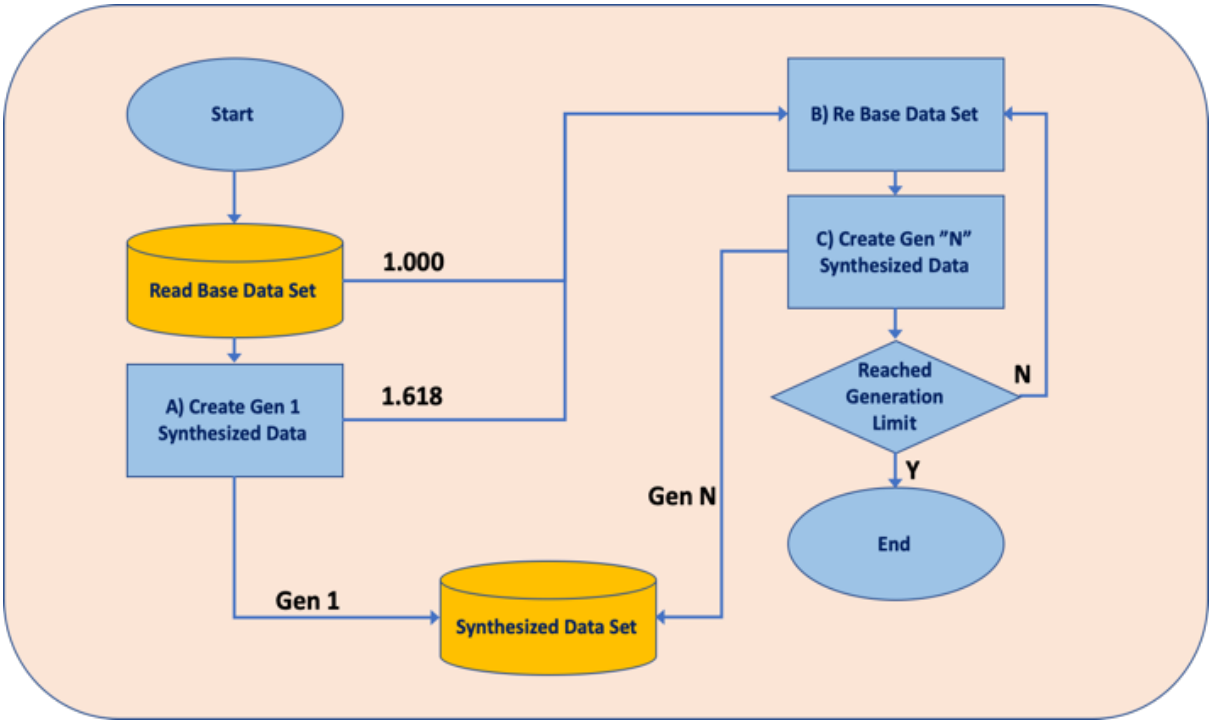
restricted based on the formula shown below. For example, the min value for the synthesized data set would be the min of the original data set minus 20% of its standard deviation.
    - *Min Value = Min (Base Dataset) – 0.2 * Standard Deviation (Base Dataset)*
    - *Max Value = Max (Base Dataset) + 0.2 * Standard Deviation (Base Dataset)*
- Check 2: The generated data set should have a similar distribution pattern as the base dataset. For each generated point, its corresponding bin is identified. If the synthesized dataset already has the same number of data points within the bin as the original set, the point is rejected. Rejecting the point ensures the generated data set is limited to the same number of values as the original data set while also exhibiting the same frequency pattern.

E. Full Data Set Created: This step validates whether the total number of generated values are equal to the number of values in the original data set. Once every bin is full, meaning it has the same amount of values in each bin as the original dataset, it passes through this check.
F. Align Generated Data Set: In the previous step the program has generated a collection of data values. However, it is not in a particular order along a timeline. As a first step, the base data and the generated data are sorted from least to greatest value. Then, each value of the generated data is assigned the time index value of its corresponding base data value.

Multi-Generation Data Synthesizer: The workflow for the Multi Gen Data Synthesizer program is shown below. This model was a modified version of the First-Generation Data Synthesizer. The workflow steps include:

A. Create Gen-1 Synthesized Data: This is where the First-Generation Model ends

B. Re-Base Data Set: A new Base Data Set is created. The Golden Ratio is used as a marker, where the new base data set is a combined result of the previous Base Data Set (the grandparent) and the newly generated data (parent) in the ratio of 1:1.618.



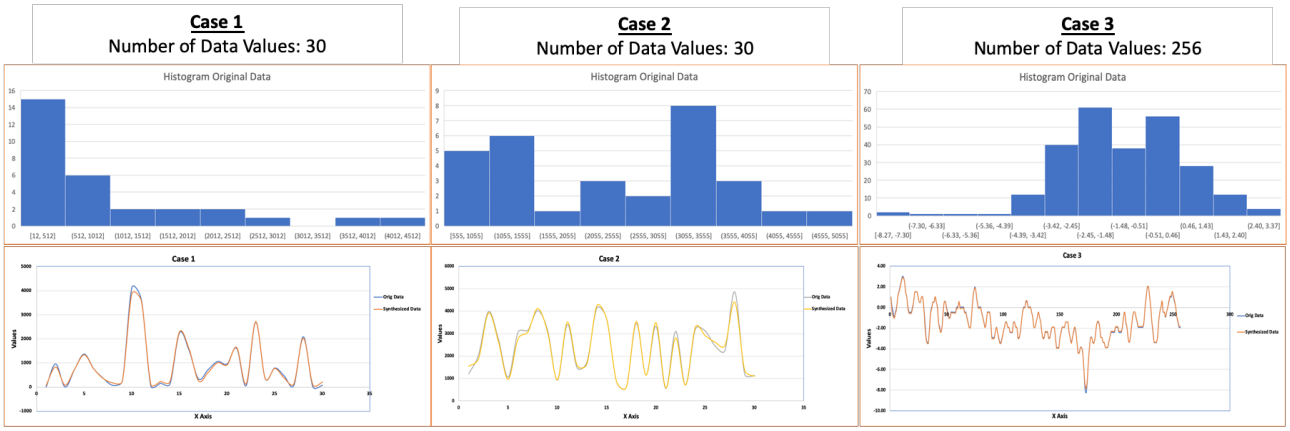
C. Create Gen "N" Synthesized Data: The First-Generation Data Synthesizer algorithm is run again, now with the new base dataset replacing the first.
D. Reached Generation Limit: If the number of generations specified has been reached, then the final synthesis is outputted. If not, the latest synthesized data and its base set are used in step B.

Program Execution Performance Study: This program was run on a 2014 Mac Mini. This is an average desktop with 8 GB Ram. For Scenario 2, this program took 2.441 seconds. However, during this execution only the 10th, 100th and 1000th generation was saved as an output.

We decided to analyze the performance further with a larger data set. We used the data set from Case 3 (Analysis of First-Generation Data Synthesizer), which was a time series of 256 data values. During this test, the program took 12.7 seconds to generate 1000 data sets.

Analysis for First Generation Data Synthesizer

The objective for this test was to analyze how quickly the synthesized data is generated for a variety of data distribution. The three scenarios that were analyzed are shown below.



	Original Data	Synthesized Data
Mean	926.97	923.30
Std. Dev.	1079.95	1037.95

	Original Data	Synthesized Data
Mean	2392.63	2378.80
Std. Dev.	1251.15	1211.75

	Original Data	Synthesized Data
Mean	-1.345	-1.349
Std. Dev.	1.752	1.736

These three cases were selected because they exhibited varied distribution. The histogram of the original data sets for each of the cases is shown in the figure above. Case 1 and Case 3 are skewed data sets while Case 2 is a representative bi-modal data set.

In all three scenarios, the program converged to a generated data set very quickly. Although in Case 3, the data set was fairly large (real world experimental data), the program converged to a solution under a second.

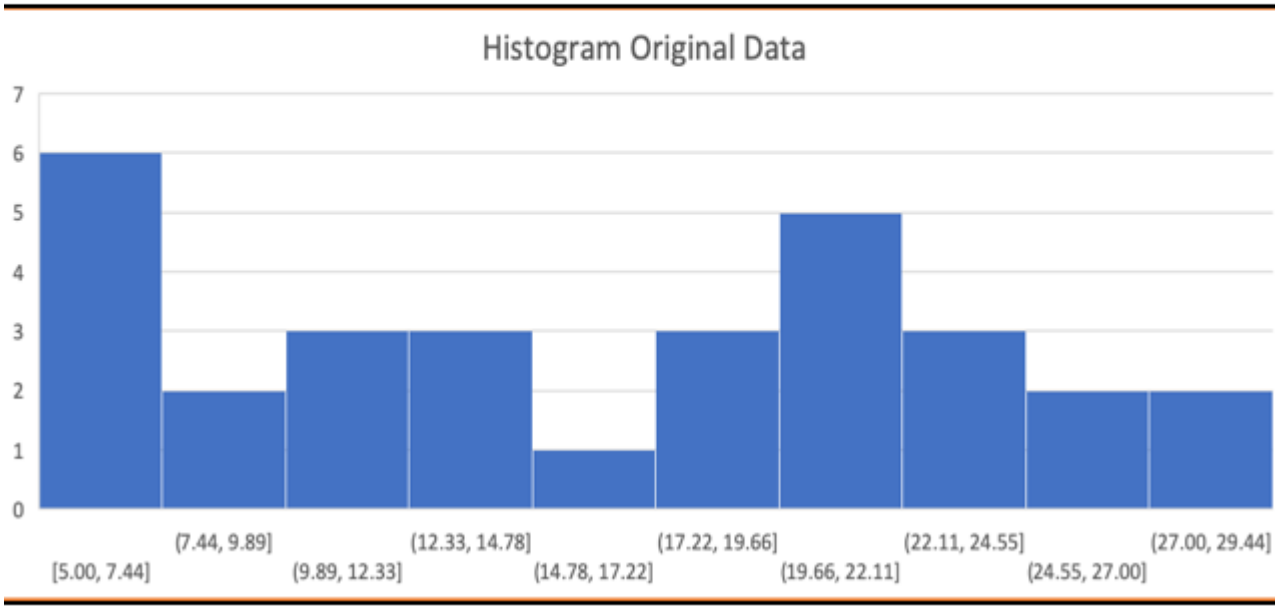
The synthesized data shows a fairly close resemblance to the original data. As discussed earlier, the generated data values are randomly generated but were a function of the mean and standard deviation of the original base data. Each of the values closely resembled that of the original data, meaning their similarities were high between the two.

Analysis for Multi Generation Data Synthesizer

This study also analyzed how the generated data would evolve over multiple generations. Here, the Golden Ratio of 1.618:1.000 was used as the basis for inheriting the characteristics of the parent vs the grandparent data set. For example, the first original data set is used to generate the first synthesized dataset. Now to create the next generation of synthesized data, it would require a new base data set. This base data set is created combining the first gen data and the original data set in the ratio of 1.618:1.000. Similarly, for each following iteration, we first create the base as discussed and generate the next data set.

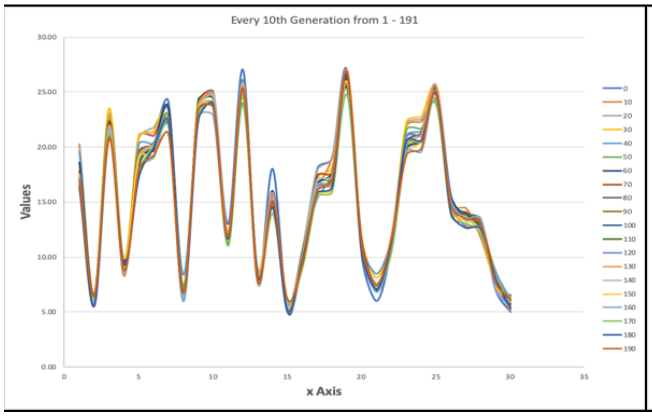
Data Distribution

Number of Data Values: 30



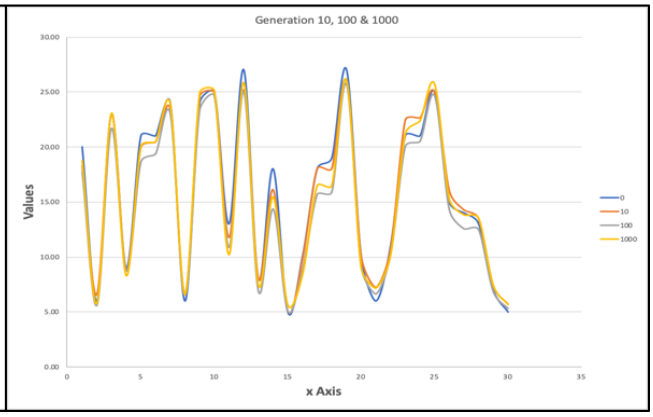
Scenario 1

Number of Data Values: 30



Scenario 2

Number of Data Values: 30



The scenario used for this test was based on the same original data set, and the distribution of the data set is shown above. In Scenario 1, the results are shown for every 10th generation until the 190th. It was taken further in Scenario 2 to analyze extreme numbers of generations, with 1000 iterations. In the above plot for Scenario 2, we show the graphs for the 10th, 100th and 1000th generation as compared to the original.

It is found that with over 1000 generations that were developed, the generated data maintained a similar profile to the original data. Most of the deviations were observed when there was a significant change in the slope of the curve.

Discussion

The algorithm created in this project can be used for numerous future applications. The algorithm's importance lies in its ease of use and accessibility. While synthetic data generation's applications are numerous and far-reaching, its use is often limited by the requirements of a powerful computer for dataset generation. Synthetic data also can be used instead of original data to avoid privacy concerns that come with real data usage. The algorithm used would be efficient, have greater usability, and allow for a more accurate analysis of data by researcher, with applications in numerous fields that require data synthesis.

Conclusion

The algorithm developed for data synthesis generation was created, tested, and the results were analyzed. With the algorithm created, the goal was to generate synthetic data set that would be randomly created, but in doing so not to exactly copy the data so statistical analysis can repeat the same results. It was concluded that the program developed during this work is an easily accessible way to generate data. The developed algorithm converges and runs quickly as tested for up to 1000 dataset generations. The program was tested with multiple time-series datasets, including real-world experimental datasets, with a variety of distribution patterns, and all tests gave expected results. It was found that larger differences in data occurred at the relative peaks & troughs of the datasets. This was an outcome of the algorithm's methodology where the sparsity of data within a bin at peaks/troughs can increase the variability of the outcome. It was also found that when less bins were used in the algorithm, the synthesized data would have greater differences to the original data, as every bin would grow in its datapoint capacity and in turn would allow for more variation at every point in the time-series dataset. A conclusion from the multi generation synthesizer was that more generations over time did not cause greater differentiation in the data, as such it did not bring any advantage to the outcome. For future work, we would like to investigate the multi-generation data synthesizer to simulate new real-world large experimental data and perhaps add more genetic properties, to improve the current design of the algorithm.

About the Data Science Foundation

The Data Science Foundation is a professional body representing the interests of the Data Science Industry. Its membership consists of suppliers who offer a range of big data analytical and technical services and companies and individuals with an interest in the commercial advantages that can be gained from big data. The organisation aims to raise the profile of this developing industry, to educate people about the benefits of knowledge based decision making and to encourage firms to start using big data techniques.

Contact Data Science Foundation

Email:admin@datascience.foundation
Telephone: 0161 926 3641
Atlantic Business Centre
Atlantic Street
Altrincham
WA14 5NQ
web: www.datascience.foundation