Feature Extraction for Time Series Data: an Artificial Neural Network Evolutionary Training Model for the Management of Mountainous Watersheds.

Thomas J. Glezakos[1], Theodore A. Tsiligiridis[1], Lazaros S. Iliadis[2], Constantine P. Yialouris[1], Fotios P. Maris[2], Konstantinos P. Ferentinos[1]

[1]*Agricultural University of Athens, Department of Science, Laboratory of Informatics, Iera Odos 75, 11855, Athens, Hellas, tsili@aua.gr*

[2]*Democritus University of Thrace, Department of Forestry & Management of the Environment & Natural Resources, Pantazidou 193, 68200 Orestiada, Thrace, Hellas, liliadis@fmenr.duth.gr*

Abstract

This manuscript is the result of research conducted towards the production of meta-data to be used as inputs to neural networks. It is essentially a preliminary attempt towards the use of an evolutionary approach to interpret the significance which time series data pose on the behavior of mountainous water supplies, proposing a model which could be effectively used in the estimation of the average annual water supply for the various mountainous watersheds. The data used for the training and testing of the system refer to certain watersheds spread over the island of Cyprus and span a wide temporal period. The method proposed incorporates an evolutionary process to manipulate the time series data of the average monthly rainfall recorded by the measuring stations, while the algorithm includes special encoding, initialization, performance evaluation, genetic operations and pattern matching tools for the evolution of the time series into significantly sampled data.

Keywords: Genetic algorithms; Artificial neural networks; Maximum volume of water flow; Average annual water supply; Evolutionary time series processing; Genetic ANN training.

1. Introduction

Cyprus, the third largest island in the Mediterranean Sea after Sicily and Sardinia, is geographically situated in the eastern Mediterranean, south of the Anatolian Peninsula of the Asian mainland. It is situated among Hellas to its west / north-west side, east to Syria, Lebanon and Israel and north to Turkey. Although commonly referred to as part of the Middle East, it is closely alligned with Europe (http://www. moi.gov.cy/pio).

The landscape, mostly mountainous at the highest percentage, includes the central plain of Mesaoria, with the Kyrenia and Pentadactylos mountains to the north and the Troodos mountain range to the south and west, while there are also scattered, but significant, plains along the southern coast. The climate is temperate Mediterranean where dry summers follow variably rainy winters. Summer temperatures range from warm at higher elevations in the Troodos mountains to hot in the lowlands. Winter temperatures are mild at lower elevations, where snow rarely occurs, but are significantly colder in the Troodos mountains. During the recent years the dry seasons are intense, rendering the lack of drinking water as a high pressure on the citizens. On the other hand, floods, lower temperatures and torrential rains throughout the wet season constitute an unusual picture for this time of year for the island (www.hri.org/news/cyprus/riken/2001/-09.riken. html, www.mediterraneangardensociety.org), causing erosion problems and destroy settlements and infrastructure. There is no doubt whatsoever that proper and efficacious water management is the key factor not only for the well being of the citizens and the satisfaction of their daily needs, but for the achievement of sustainable development as well.

The primary aim of the present survey, a successor to previous research work conducted by Iliadis and Maris [6], is to present a preliminary attempt towards the design and implementation of an evolutionary technique in the line of producing meta-data for the training and testing of Artificial Neural Networks (ANN). The meta-data were produced out of raw time series data, in the hopes of eliminating the noise inherent in the initial data, in order to develop a highly adaptive evolutionary model towards decision making

in water resources management. The genetic algorithm governing the meta-data production performs a forward crawl through the input space in search of combination of genes which outperform their brethren after they have been used as inputs to the neural network, the performance of which is evaluated by means of its root mean square error. The network performs an effective estimation of the Average Annual Water Supply (AAWS) on an annual basis, for each mountainous watershed of Cyprus, since the estimation of this factor plays a highly important role to the management of mountainous water resources, as it is closely related to the mountainous watershed fermentations, as well as to the potential torrential risks [5][12][13][14]. The input data consist of factors which, according to their type may effectively be classified into two categories: structural data in the sense that they remain constant variables for the whole time span of the research, as opposed to dynamic data, category which holds ever changing variables.

The parental research of Iliadis and Maris used three structural input parameters, namely the area of watershed, the altitude and the slope and two dynamic ones, the average annual and the average monthly rain height. For the time span of each dynamic factor, the monthly measurements had been averaged and the result was used as input. The research key point was that it used only two dynamic parameters and among them, only the rain height should be monitored monthly and annually, rendering the acquisition of model inputs effortless and inexpensive both financially and in human resources. The current research differs in that it does not utilize the average annual rain fall; instead, we have elected to bring the whole monthly average measurements into play and to search for the best combination of measurements as inputs to the neural network. The key feature of this attempt is that while it keeps the data acquisition at low level of expenses, it also goes to a lower level of input data search, essentially rendering the input vector more precise. Once trained, the proposed ANN continues to be highly adaptable to various regions and places, provided that the inputs will have been manipulated according to the genetic algorithm decisions.

1.1 The ANN Concept.

The human brain, is a highly complicated biologic machine, capable of solving innumerable kinds of problems, from the most simplistic to highly complex ones. Artificial Neural Networks (ANN) were developed in an attempt to simulate the function of the human brain. An ANN is a software device consisting of a number of simple processing elements interconnected and operating in parallel. Each neuron is only aware of the signals it receives from other connected neurons and the information it sends from time to time to other processing elements. In this context, an ANN is a computer program capable of learning from examples through iteration. In most of the times no prior knowledge of the input data is required, because the training process is essentially a search for the best synaptic weight vector. Learning is the process of adapting or modifying the neurons' connection weights in response to stimuli presented as inputs requiring the presence of a known output. This process enables the network to learn to solve problems by adequately adjusting the strength of the connections between their processing elements according to the input data and the desired outputs. ANNs have been vastly used for recognizing patterns in the input data space or to extract simple rules for complex problems according to their inputs. Key factor in every neural network training is generalization, the capability of the network to predict "unseen" inputs merely with the knowledge that has been acquired during the training process, in which a stimulus presented in the output corresponds to a desired response for a given input.

The ANN concept is currently widely used in various research works, ranging from pattern recognition, quality control, classification and have gained wide recognition in modelling many processes in engineering [7][15][10][4][11][2]. Lately they have been used to predict wood water isotherm or sorption isotherms in food science [1][9] and numerous other disciplines. A neural network may learn by example and outmatches rivalling techniques in that it may use its knowledge under untrained circumstances incorporating a large number of variables [3].

1.2 A Brief Reference to the Genetic Algorithms Concept

Genetic algorithms are inspired by evolutionary biology, and especially driven by the Darwinian axiom of the "survival of the fittest", incorporating numerous biological procedures such as inheritance, selection, crossover (or recombination) and mutation. They are mostly implemented as computer simulations which search for better solutions to a given optimization problem. The genetic algorithm starts out with an, often random, initial population of encoded representations of candidate solutions. These representations are referred to as chromosomes, genotypes or genome, while the candidate solutions are referred to as phenotypes. The algorithm proceeds by generations each of which is comprised by genotypes of the previous generation, which are elected basically by their fitness and modified on the grounds of a possible recombination or mutation to form a new population of higher overall fitness. An overview of genetic algorithms and their implementation in

computer science / machine learning may be found in http://www.doc.ic.ac.uk/~nd/surprise _96/journal/vol4/tcw2/report.html#SHOP.

2. Necessity for the Genetically Trained ANN Development

It is beyond doubt that an innovative approach may come in handy, especially considering the fact that after time – consuming studies and raw data acquisition, the Republic of Cyprus did not come up with a viable solution to the problem of water resources management. According to Iliadis and Maris [6], the classical statistical analysis failed to support this solution, although newer attempts have been conducted towards this purpose. The recent findings of the undergoing research have revealed that the water resources of the island are much lower than it was initially regarded. The correct number of the island's water reservoir is now considered to be approximately 40% lower than the original belief. This fact urges for a new and perhaps more reliable approach.

3. Materials and Methods

3.1. Research Area and data acquisition

The research area, as in the initial research [6], covers all of the mountainous watersheds which are under the administration of the Republic of Cyprus. Specifically, the island is divided into nine water Divisions with a total of seventy Torrential Streams. As already noted, the two most important landscape characteristics of the island are the Kyrenia and Pentadactylos mountains to the north with an altitude which reaches 1000 m. and a length of 160 km., and the Troodos mountain range to the south and west with a maximum altitude of 1951 m.

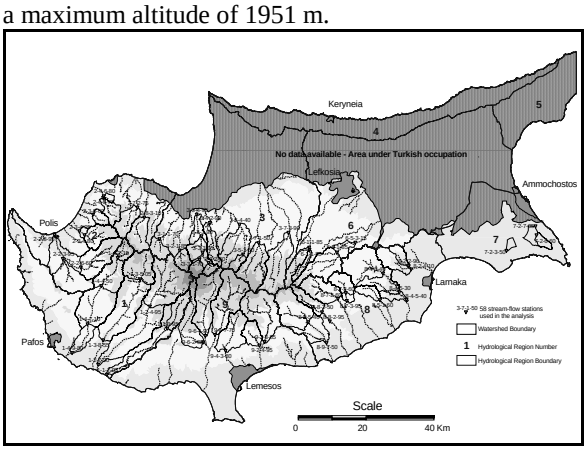


Figure 1. General view of Cyprus mainstreams

The initial dataset accumulated out of 78 stations located at the span of the seventy torrential streams. The time span of the initial data set covered a period of 28 years, from 1965 to 1993, for most of the stations' measurements. The current research though, did not average the rainfall time series, but rather took under consideration the average for every month for each year. Figure 1 depicts a general view of the island's mainstreams, while Figure 2 presents the average annual rain height (mm) from 1970 to 2000. Table 1 on the other hand presents a small indicative sample of the data used. All the measurement stations along with the data which they have collected for the whole time span of the research, belong to the Ministry of Agriculture Natural Resources and Environment of Cyprus [8].

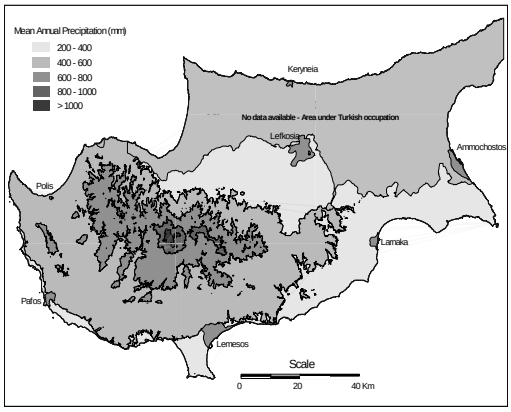


Figure 2. Average annual rain height (mm) from 1970 to 2000.

Table 1. Sample of the gathered data

	Pattern 1	Pattern 2	Pattern 3
Station Code	1113 614 95	1117 898 95	1142 915 15
Year	1965-66	1965-66	1979-80
M1	229	201	195
M2	62	67	165
M3	89	77	135
M4	13	8	25
M5	4	4	9
M6	0	0	0
M7	0	0	0
M8	0	1	5
M9	70	54	1
M10	146	102	60
M11	23	19	133
M12	143	146	169
Fn (km^2)	38	110	22
Qmax	54	120	8.6
P(mm)	770	660	810
H (m)	550	8	600
Jk (%)	4.2	2.2	8
Qmy(m3/s)	420.62	611.95	330.6

where M1 – M12: the months of the year, Fn: Area of Watershed, Qmax: Maximum supply, P: Average annual Rainfall, H: Absolute altitude, Jk: Absolute slope, Qmy: Average annual water supply.
In this context we accumulated a range of 1411 records covering the aforementioned period of time. It was essential for our code to remove any record which had missing values in one or more of the studied factors, so this procedure left us with 1273 patterns of data (inputs and outputs together). This initial recordset was used for the formulation of the training and the testing data sets, comprising of 1152 and 121 patterns respectively. The inputs to the system were the area of the watershed (in km^2), the absolute altitude (in meters), the absolute slope (in percentage) as structural input data, whereas the dynamic input data consisted of the maximum water supply, the average annual rainfall, as well as the average rainfall for each month for each year of measurement. The network had only one output, the Average Annual Water Supply (in m^3/s)

3.2 The Python Programming Language and the Fast Artificial Neural Network Library (FANN)

The research team chose to develop code with the much promising language of Python. This selection was two-fold propulsed: on one hand to gain insight in one of the most rapidly developing and integrated computer programming languages available today and, on the other, to use freely distributable open source products. Python, as proved out to be, is an easy to learn, yet powerful programming language, incorporating efficient high level data structures as well as a simple yet effective approach to object oriented programming. Being a multi platform language, it allows for programs to be developed on most of the platforms available today. Our source was writen in Ubuntu Linux, but can be run on a windows – based system just as easily. Python is an interpeted language. This, of course, renders it slower than any compiled language, but the robust approach it offers more than compensates for it. Being an open source project, Python is supported by a vast number of freely available libraries, in addition to the extensive embedded library of its own. In addition, the interpreter is easily extended with new functions and data types implemented in C or C++, standard objects and/or modules.

Python, as a dynamically developed programming language, is supported by a vast number of freely distributed libraries. One of these is the Fast Artificial Neural Network library (FANN), a free open source neural network library implementing multilayer artificial neural networks in C, with support for both fully connected and sparsely connected networks, while cross-platform execution in both fixed and floating point are supported and includes a framework for easy handling of training data sets. The FANN library proved out to be easy to use, versatile, well documented and fast and was used by the research team so as to construct the neural network model of the application. The evolutionary process was coded in Python from scratch, while the neural network was embedded into it, in order to provide the fitness for the population in each generation and to contribute in the selection of the fittest chromosome.

4. The Algorithm

Our research in its essence was to develop a commonly used procedure for the production of meta-data from time series, so the implementation of an evolutionary process, specifically a genetic algorithm, seemed as an one-way road. Roughly the algorithm should be able to produce an initially random population of ‘trainers’ in its initial generation, that is chromosomes behaving in certain ways and able to appropriately manipulate the initial raw time series. The products of the trainers should be assigned a ‘fitness’, that is a floating point value from some function. In our case the fitness of each evolutionary produced time series was the root mean square error of a neural network. Thereinafter, the next and subsequent generations of the algorithm should be formulated according to a selection policy which should elect the fittest members of the previous generation of trainers.

4.1 Initial Data Manipulation

The application starts out by incorporating the initial training and testing data set, and then analyzing it in order to create the initial random population of trainers / chromosomes. It collects the initial training and testing data and saves them to a locally created nested list, each sublist of which corresponds to one line of the initial raw data time series train and testing files. Consequently, the data is analyzed and the portion of it that corresponds to the initial time series (in our case the twelve values corresponding to the rainfall averages for the months of each year) will be used for the creation of the basic training population of the algorithm, while the rest of the initial data, which correspond to the structural and dynamic input non time series data are set aside for future use.

The initial generation of the algorithm starts out with a user defined number of training chromosomes, which comprise the basic trainer. Each chromosome of the trainer contains twelve randomly chosen bits in the range of [0, 1], plus one bit in the beginning of the

chromosome which stands out as a mechanism bit, driving and manipulating the behavior of the whole chromosome. So the chromosomes of the trainer for each generation is a binary list of *n+1* elements each, where *n* is the initial time series data (twelve for our case) and the excess gene is the "mechanism gene" of the chromosome. The function of the trainer is to essentially "map" its genes to the initial training and testing data sets, according to its mechanism gene. If this is 0, then it stands out as a "Discard-All-Zeros" resampling function. In this case, the time series will be stripped off of its values for which the corresponding genes of the trainer is 0. On the other hand, if it is 1, then it stands out as a "First-One-Last-Zero" clustering mechanism for the initial data. In this last case, the trainer extracts the everage of the time series elements for every group of its own genes which start with the first 1 and end with the last zero.

It is essential to be cleared at this point that the trainer of the initial generation of the algorithm explicitly contains a chromosome having all of its genes equal to 1, so as to bring forth and test the initial time series unaltered, along with all other tests conducted.

Following the mapping of the trainer chromosomes to the time series training and testing data, is the joining phase, in which each "genetically" produced time series training list will again embed the structural and dynamic data from which was initially deprived, in order to be genetically evolved. In the next phase neural training and testing comes into play.

4.2 Neural Training and Testing

In the beginning we confronted the dilemma of constructing a new neural network structure from the ground up, making a variety of testing and trial and error procedures, or to follow the initial research and test the algorithm against the original findings. We chose to proceed with the latter solution, so as to be able to compare the results on more steady grounds. The only alterations which we were unable to avoid were the changing in the input layer of the neural network. That happened because each trainer produces different kind of data of varying inputs, although the output remains unchanged. The code was enhanced with a neural network object from the FANN library, which was used as a landmark for the fitness of the chromosomes. The Neural Network object was created and trained as a standard backpropagation Multi Layer Perceptron (MLP), with 3 layers: one input layer with varying neurons (according to the input training and testing file), one hidden layer of 9 neurons according to the initital research and, finally, an output layer of one neuron, predicting the average annual water supply. The fact that a neural network may perform well on its training data does not necessarily ensure that is a good module. The only positive indication that a network performs well is its generalization capabilities. As generalization we conceive the ability of the neural network to predict correctly on new 'unseen' data. We utilized the standard way to test an ANN in our research by setting aside a number of initial training patterns to formulate the testing data set [3]. The fitness function implemented later in the genetic algorithm takes under consideration the testing error of the network, the error that is derived by patterns which were not used at all during the training phase.

The neural object for the application returns the Root Mean Square error for the testing data each time.The input layer of the neural network object was designed so as to be auto – formatted according to each training and testing file that came in turn. This was implemented via the *create_standard_array()* function of the FANN library which, among others, permits for such a behaviour of the network. We should note at this point that before proceeding to the formulation of the algorithm, we conducted a series of tests in order to verify that the <inputs>/9/1 structure of the network was an acceptable choice for our own data. The series of tests was realized by the *cascadetrain_on_data()* function of the library. The module of cascade training is totally different from ordinary training, permitting the network to start out empty in the hidden layer. Then, as training starts and continues, it adds neurons one by one and layer by layer, until an optimal neural network structure is reached (http://leenissen.dk/fann). For each neuron added we tried several activation functions and training algorithms. Table 2 shows the activation functions tried.

Table 2. Activation Functions used

Activation Function	Dependent Variable Span	Description
Linear	-inf < y < inf	y = x*s, d = 1*s
Threshold		x < 0 -> y = 0, x >= 0 -> y = 1
Sigmoid	0 < y < 1	y = 1/(1 + exp(-2*s*x)) d = 2*s*y*(1 - y)
Sigmoid Symmetric	-1 < y < 1	y = tanh(s*x) = 2/(1 + exp(-2*s*x)) – 1 d = s*(1-(y*y))
Gaussian	0 < y < 1	y = exp(-x*s*x*s) d = -2*x*s*y*s
Gaussian Symmetric	- 1 < y < 1	y = exp(-x*s*x*s)*2-1 d = -2*x*s*(y+1)*s

Where x is the input to the activation function, y is the output, s is its steepness and d is the derivation (http://leenissen.dk/fann).

The training algorithms we tried include Incremental Training, Batch Training and the popular Rprop algorithm. The incremental training constitutes the basic standard backpropagation algorithm, where the weights of the neural network are updated immediately after each trianing pattern is shown to the network, producing a numerous weight updating during a single epoch of training. Batch training on the other hand, is implemented by updating the weights once after the epoch has been completed, that is after the root mean square error has been calculated for the whole trianing set. This category includes both the simple batch training and the rprop training algorithms.

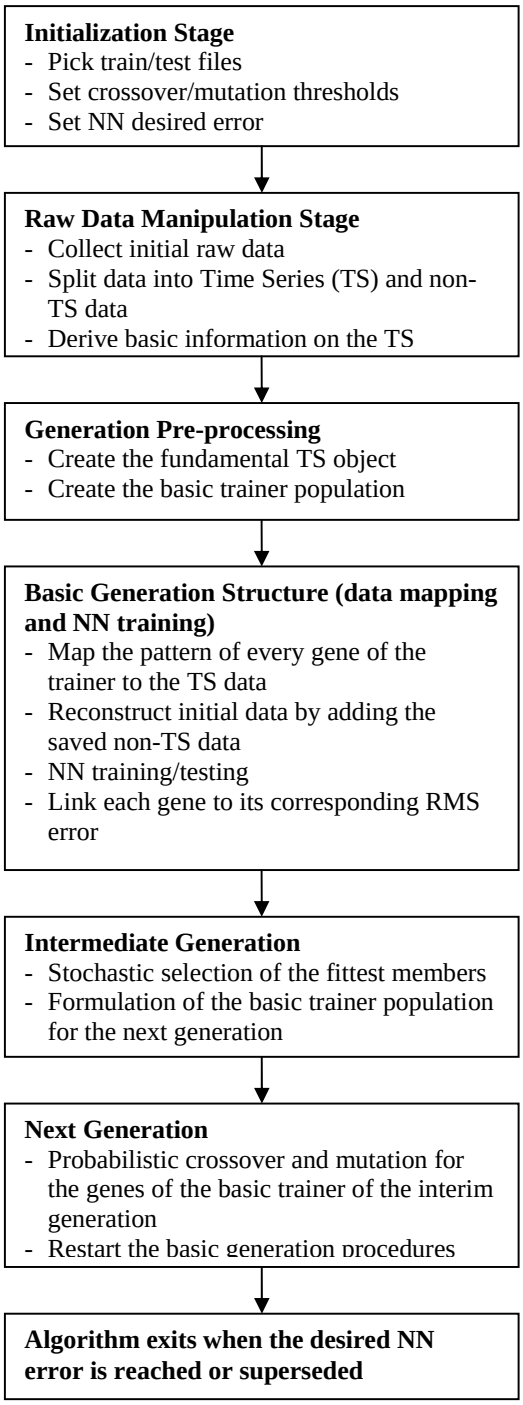
The best results were achieved with the Rprop training algorithm in combination with the Sigmoid Symmetric (hyperbolic tangent) activation function of the neurons. This procedure confirmed that the aforementioned proposed structure was acceptable for most of the cases.

By performing the neural network trining for all the evolutionary produced time series data and acquiring the network root mean square error for each, we assigned the RMS error to the corresponding trainer. Having initially set an ideal desired neural network error, the distance of the recorded error from the ideal already set, should suffice to stand as the fitness value for the corresponding chromosome. The algorithm proceeded in the selection of the fittest chromosomes for the next generation.

4.3 Selection Policy and Intermediate Generation

In order to formulate each next generation of the algorithm, there was a need to create a 'behind-the-scenes' intermediate generation at the end of the current generation. This holds the fittest members of the generation, as well as other chromosomes not so fit. The policy for the selection incorporated a stochastic procedure. In order to send a copy into the intermediate generation, a chromosome should have fitness under a user defined arbitrary value. Moreover, there was also a test for the real RMS value for every chromosome: if the RMS value was smaller or equal to a randomly produced real number in an arbitrary range, then a second copy of the chromosome should be copied to the intermediate generation. The intermediate generation should be essentially a set of trainers with population twice as much as the original trainer population. If after the application of the aforementioned selection policy the population of the intermediate generation was not the desired one, the algorithm continued to copy the fittest members of the original trainer population.

The following diagram graphically illustrates the evolutionary training procedure.



4.4 Formulation of the Next Generation

At the point when the population of the intermediate generation was fixed, the genetic

mechanisms of the algorithm formulate the next generation. For each randomly picked pair of chromosomes of the intermediate generation there is an arbitrary set probability of selection, recombination (crossover) or mutation. Recombination of chromosomes is the procedure where the parents contribute with different supplementary parts of their genome in the production of their offspring. In our algorithm, the 'breaking point' for the chromosome of the parent is random within the bounds of the parent genome. On the other hand, mutation refers to the 'flipping' of an offspring's random gene.

It is well known that, as for all the machine learning problems, a proper amount of time should be invested in the fine tuning of the mutation and recombination probability of the genetic algorithm. An excessively small mutation rate may lead to "genetic drift", that is the statistical effect that stems from the influence thatprobability poses on the survival of alleles (variants of a gene, 0 or 1 in our case) and the trait that it confers to the chromosome. A positive genetic drift renders the allele paramount in the genetic pool, whereas a negative genetic drift may extinct the allele. Both limits, either too high, or too low, in the genetic drift could potentially pose irreparable damage to the genetic pool. The same holds true for the crossover probability. It is a fact that a variety of crossover and mutation probabilities were tested, concluding on selecting 0.4 and 0.005 respectively.

5. Results

For every genetic algorithm the aforementioned evolutionary process continues until a stopping condition is met. The most common terminating conditions include the satisfaction of the minimization (or maximization) criteria by one or more generations, the trapping of the generations to a minimal plateau which is not improving by successive generations any longer and, finally, if a fixed number of generations has been reached.

The algorithm implemented in this research falls under the last category of stopping conditions, mainly due to computational power constraints. Instead of letting the developed algorithm run for a vast number of generations until reaching a solution plateau, we chose to allow it to run for five generations, while the whole procedure was repeated for five times, in order to have a better understanding on the performance of the search with various initial starting points. The results are illustrated in Table 3 and on Figure 3, where it is made clear that the evolutionary process performs well, crawling steadily to an optimal solution.

6. Discussion

It has been made obvious by recent trends in climate change that the management of water resources is of great importance, especially for a region such as Cyprus which has already bear a significant amount of environmental pressure. The estimation of the Average Annual Water Supply plays a very important role to the said water resources management, as it is closely related to the mountainous watershed fermentations on one hand and to the potential torrential risks on the other. It is of great importance for the policy makers to have reliable integrated tools at their disposal, tools which could inform them about the course of the phenomena in the near future. It is also of great importance that these tools requirements are kept at as lower cost as possible, both at the financial level and at the human-day occupation.

The main advantage of the present research work, as well as its parental one, is the fact that it proposes the development of a tool for the prediction of a key factor in water management and torrential risk and elimination, which requires minimal effort and expense as it measures only two dynamic input factors. Obviously, the structural input data are not to change for the relatively small era of measurements that are conducted, thus the only factor which is to be monitored as precisely as possible on a daily basis is rain - height. Furthermore, the developed module can be re-adjusted and developed by continuous training, as new input data flow in and, provided the availability of training data could be proposed for different regions.

The crucial contribution of the present research work is the evolutionary clustering and re-sampling of the initial time series data. The advantages of the proposed approach are numerous. Firstly, the researcher and the developer of the system has a vast number of training / testing data at his disposal, because now we do not need to average the whole time span of measurements of the monthly rain height and create only one record of input patterns. Instead, every year has become an input pattern for our system, effectively multiplying the number of the initial data set. Also, the evolutionary process diminishes both the total number of inputs, as well as the potential noise inherent in the input time series data. This way we have succeeded in effectively train the proposed neural network and produce reliable estimations, while keeping the operating costs at acceptable levels.

There is always, of course, the invariable dilemma of the representation of the gene. In our case, we have conducted a number of training scenarios some of which included the floating point representation of the basic trainer genes, instead of the binary one. This configuration though posed a lot of problems and was eventually dropped. For one, the whole system was

very demanding in computing power, partly due to the requirements posed by the programming language, as well as the structure of the initial row data itself. The system also crashed in certain circumstances, when the algorithm produced illegal gene values and furthermore illegal clustering configurations in the time series.

Although these obstacles were too important to be overlooked, the benefits committed by a floating point representation of the gene, such as greater variability among the offspring and increased probability for the algorithm to overcome trapping in luring local minima, has posed a challenging prospect for the future. The plans of the research team is to essentially widen the scope of the algorithm, along with its potential, by adding more machine genes at the chromosomes of the generation trainers, so as to force them to expand their search plane. This development may probably require the re-engineering of the core code of the program so as to keep its computational demands at as low a level as possible. Also, the development of a Graphical User Interface (GUI), which could render the application friendlier to the average user, could contribute in the acceptance of the software to different regions.

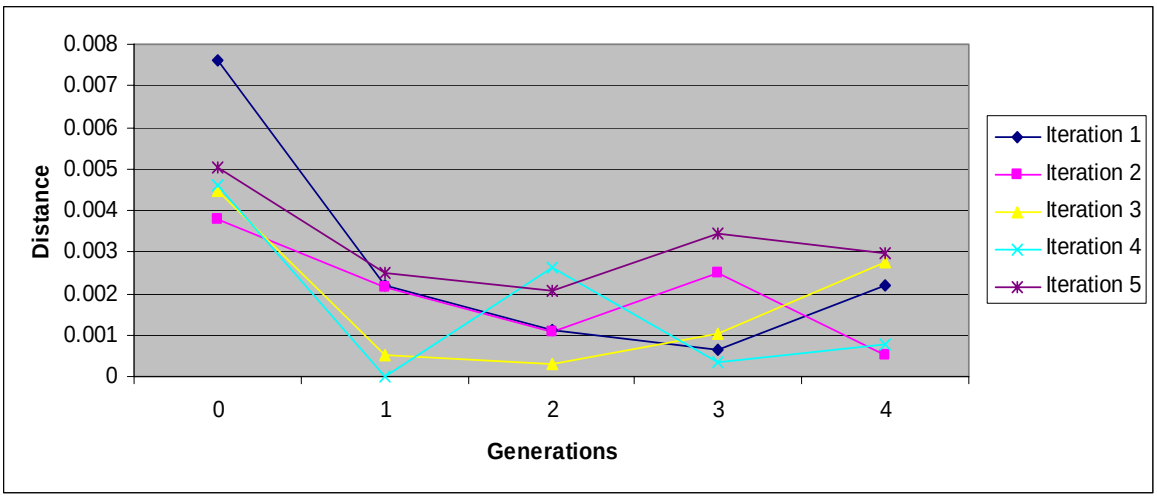


Figure 3. The course of the evolutionary process

Table 3. Results of the evolutionary process

Iteration	Generation	Chromosome	RMS Error	Distance from set Error
1	0	1100111010011	0.0106134622	0.0076134622
1	1	1111011001101	0.0121847362	0.0021847362
1	2	0100001000011	0.0089017823	0.0010982177
1	3	0100010001001	0.0093576869	0.0006423131
1	4	1100010001001	0.0122147424	0.0022147424
2	0	1100000000000	0.0061973166	0.0038026834
2	1	0100001000011	0.0078532060	0.0021467940
2	2	0100001000011	0.0089220124	0.0010779876
2	3	0100001000011	0.0075098990	0.0024901010
2	4	1101001011111	0.0105336589	0.0005336589
3	0	1100000000000	0.0055064529	0.0044935471
3	1	1101001101101	0.0105042536	0.0005042536
3	2	1101001001001	0.0102993900	0.0002993900
3	3	0100001001001	0.0089830988	0.0010169012
3	4	1100000100111	0.0127581959	0.0027581959
4	0	1100000000000	0.0053980419	0.0046019581
4	1	1110100110101	0.0100132720	0.0000132720
4	2	1101000110101	0.0126448840	0.0026448840
4	3	1101001101001	0.0103567051	0.0003567051

4	4	0111100100101	0.0092294285	0.0007705715
5	0	1100000000000	0.0049562710	0.0050437290
5	1	1100011000001	0.0124807614	0.0024807614
5	2	1100000000001	0.0079311285	0.0020688715
5	3	1100001010001	0.0134370692	0.0034370692
5	4	1100000000001	0.0070299043	0.0029700957

References

1. Avramidis, S. and L. Iliadis, "Wood-Water Isotherm Prediction with Artificial Neural Networks: a Preliminary Study" Holzforschung. ISSN: 0018-3830 59 (3), 336-341. Berlin, New York, Walter De Gruyter & Co.
2. Boillereaux L., Cadet C. and A. Le Bail, 2003. Thermal properties estimation via real time neural network learning. J. Food Eng. 57:17-23.
3. Haykin, S., 1999. Neural Networks, A Comprehensive Foundation. Prentice Hall International, New Jersey.
4. Hussain M.A., Safiur Rahman M. and C.W. Ng., 2002. Prediction of pores formation (porosity) in foods during drying: generic models by the use of hybrid neural network. J. Food Eng. 51:239-248.
5. Iliadis, L., Spartalis, S., 2005. Fundamental Fuzzy Relation Concepts of a D.S.S. for the Estimation of Natural Disasters' Risk (The case of a trapezoidal membership function), Mathematical and Computer Modelling, Elsevier, 42, p.747-758 (2005).
6. Iliadis, L.S. and F. Maris, "An Artificial Neural Network model for mountainous water-resources management. The case of Cyprus mountainous watersheds", *Environmental Modelling and Software*, Elsevier, 2006.05.26, pp. 1-7.
7. Jambunathan K., Hartle S.L., Ashforth-Frost S. and V.N. Fontana, 1996. Evaluating convective heat transfer coefficients using neural networks. Int. J. Heat Mass Transfer 39(11):2329-2332.
8. Kyprhs D., Neofytou P., 1998. Monhtly River Supplies of Cyprus, Monthly Rain-Falls, Maximums of instant flow. Ministry of Agriculture Natural Resources and Environment, Department of Water Development. Cyprus.
9. Myhara R.M., Sablani S., 2001. Unification of food water sorption isotherms using artificial neural networks. Drying Technol. 19(8):1543-1554.
10. Paliwal J., Visen N.S. and D.S. Jayas (2001) Evaluation of neural network architectures for cereal grain classification using morphological features. J. Agri. Eng. 79(4):361-370.
11. Sablani S.S., Baik O-D. And M. Marcotte, 2002. Neural networks for predicting thermal conductivity of bakery products. J. Food Eng. 52:299-304.
12. Spartalis, S., Iliadis, L., Maris, F., 2007. An Innovative Risk evaluation System estimating its own Fuzzy Entropy, Mathematical and Computer Modelling, Elsevier, Vol.46, Issues1-2, pp.260-267
13. Spartalis, S., Iliadis, L., Maris, F., Marinos, D., 2004. A Decision Support System Unifying Fuzzy Trapezoidal Function Membership Values Using T-Norms: The case of river Evros Torrential Risk Estimation, ICNAAM2004 Intern. Conf. on Numerical Analysis and Applied Mathematics WILEY-VCH VerlagGmbH&Co.KGaA,ISBN3-527-40563-1, pp. 173-176.
14. Spartalis, S., Iliadis, L., Unsupervised Neural Clustering applied on Fuzzy Conjunction Vectors accepted for publication in the Journal "Mathematical and Computer Modelling, Elsevier.
15. Sreekanth S., Ramaswamy H.S. and S.S. Sablani, 1998. Prediction of psychrometric parameters using neural networks. Drying Technol. 16(3-5): 825.