Evolutionary Prediction of Total Electron Content over Cyprus

Alexandros Agapitos[1], Andreas Konstantinidis[2], Haris Haralambous[2] and Harris Papadopoulos[2]

[1] School of Computer Science and Informatics, University College Dublin, Dublin, Ireland
alexandros.agapitos@ucd.ie

[2] Computer Science and Engineering department, University of Frederick, Nicosia, Cyprus
{ com.ca, eng.hh, h.papadopoulos}@frederick.ac.cy

Abstract. Total Electron Content (TEC) is an ionospheric characteristic used to derive the signal delay imposed by the ionosphere on trans-ionospheric links and subsequently overwhelm its negative impact in accurate position determination. In this paper, an Evolutionary Algorithm (EA), and particularly a Genetic Programming (GP) based model is designed. The proposed model is based on the main factors that influence the variability of the predicted parameter on a diurnal, seasonal and long-term time-scale. Experimental results show that the GP-model, which is based on TEC measurements obtained over a period of 11 years, has produced a good approximation of the modeled parameter and can be implemented as a local model to account for the ionospheric imposed error in positioning. The GP-based approach performs better than the existing Neural Network-based approach in several cases.

Keywords: Global Positioning System, Total Electron Content, genetic programming.

1 Introduction

The ionosphere is defined as a region of the earth's upper atmosphere where sufficient ionisation can exist to affect the propagation of radio waves. It ranges in height above the surface of the earth from approximately 50 km to 1000 km. The influence of this region on radio waves is accredited to the presence of free electrons. The impact of the ionosphere on communication, navigation, positioning and surveillance systems is determined by variations in its electron density profile and total electron content along the signal propagation path [1], [2]. As a result satellite systems for communication and navigation, surveillance and control that are based on transonospheric propagation may be affected by complex variations in the ionospheric structure in space and time leading to degradation of the accuracy, reliability and availability of their service. Total Electron Content (TEC) is an important parameter in trans-ionospheric links since when multiplied by a factor which

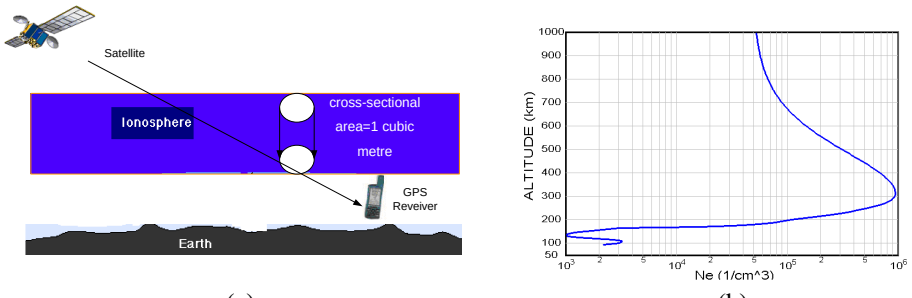
is a function of the signal frequency, it yields an estimate of the delay imposed on the signal by the ionosphere due to its dispersive nature.

This paper describes an attempt to develop a model to predict TEC above Cyprus to encapsulate its variability on a diurnal, seasonal and long-term scale. The model development is based on around 60000 hourly TEC measurements recorded above Cyprus from 1998 to 2009. The practical application of this model lies in its possible use as an alternative candidate local model to the existing Klobuchar global model [3] that is currently being used in single frequency GPS navigation system receivers to improve positioning accuracy.

Metaheuristics and more specifically Evolutionary Algorithms were proven efficient and effective in dealing with difficult-to-solve real-life problems [4]. Particularly, Genetic Programming (GP) based approaches performed well in evolving computer programs, controllers and models [5] in the past. In this paper, we have adopted a Pareto-based Genetic Programming approach for dealing with the TEC problem. The proposed GP is a panmictic, generational, elitist genetic algorithm with an expression-tree representation [5]. To the best of our knowledge this is the first time that a GP-based approach is applied to the proposed problem. The main contribution of our paper is: a GP-based prediction model designed for the TEC over Cyprus that outperforms the previously proposed Neural Network-based model [6].

2 Measurements and characteristics of TEC

Dual-frequency GPS data recorded by GPS receivers enable an estimation of the Total Electron Content (TEC) measured in total electron content units, (1 TECU = 10^{16} elecrons m^{-2}). This is the total amount of electrons along a particular line of sight between the receiver and a GPS satellite in a column of 1 m^2 cross-sectional area and represents a typical quantitative parameter of interest to GPS users (see Figure 1a). TEC is therefore the integral of the electron density profile (see Figure 1b) from the ground to an infinite height (practically the height of the satellite).



(a) (b)

Figure 1. TEC representation and typical electron density profile of the ionosphere over Cyprus.

The electron density of free electrons within the ionosphere and therefore TEC depend upon the strength of the solar ionizing radiation which is a function of time of day, season, geographical location and solar activity [1], [2]. Since solar activity has an impact on ionospheric dynamics which in turn influence the electron density of the ionosphere, TEC also exhibits variability on daily, seasonal and long-term time scales in response to the effect of solar radiation. It is also subject to abrupt variations due to enhancements of geomagnetic activity following extreme manifestations of solar activity disturbing the ionosphere from minutes to days on a local or global scale. The most profound solar effect on TEC is reflected on its daily variation as shown in Figure 2. As it is clearly depicted, there is a strong dependency of TEC on local time which follows a sharp increase of TEC around sunrise and gradual decrease around sunset. This is attributed to the rapid increase in the production of electrons due to the photo-ionization process during the day and a more gradual decrease due to the recombination of ions and electrons during the night. The long–term effect of solar activity on TEC follows an eleven-year cycle is also clearly shown as we can observe a marked correlation of the mean level of TEC and sunspot number which is an established index of solar activity.

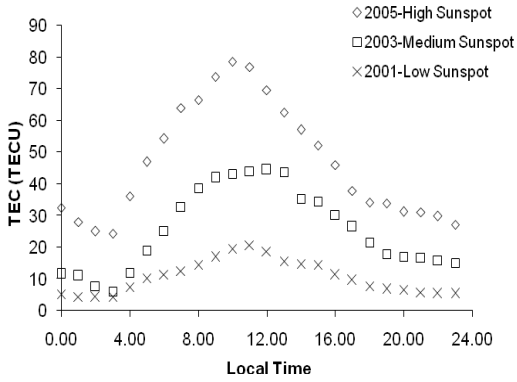


Figure 2. Diurnal variability of TEC for low, medium and high solar activity.

There is also a seasonal component in the variability of TEC which can be attributed to the seasonal change in extreme ultraviolet (EUV) radiation from the Sun. This can be clearly identified in Figure 3 for noon values of TEC for high and low solar activity periods (years 2001 and 2008).

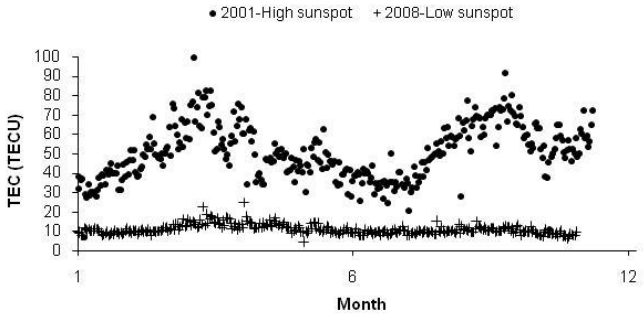


Fig. 3. Seasonal variation of TEC at 12:00

3 Model Parameters

The diurnal variation of TEC is clearly evident by observing Figure 2. We therefore include hour number as an input to the model. The hour number, hour, is an integer in the range $0 \leq \text{hour} \leq 23$. In order to avoid unrealistic discontinuity at the midnight boundary, hour is converted into its quadrature components according to:

$$sinhour = \sin\left(2\pi \frac{hour}{24}\right) \quad (1) \qquad \text{and} \qquad coshour = \cos\left(2\pi \frac{hour}{24}\right) \quad (2)$$

A seasonal variation is also an underlying characteristic of TEC as shown in fig. 3 and is described by day number daynum in the range $1 \leq \text{daynum} \leq 365$. Again to avoid unrealistic discontinuity between December 31st and January 1st daynum is converted into its quadrature components according to:

$$sinday = \sin\left(2\pi \frac{daynum}{365}\right) \quad (3) \qquad \text{and} \qquad cosday = \cos\left(2\pi \frac{daynum}{365}\right) \quad (4)$$

Long-term solar activity has a prominent effect on TEC. To include this effect in the model specification we need to incorporate an index, which represents a good indicator of solar activity. In ionospheric work the 12-month smoothed sunspot number is usually used, yet this has the disadvantage that the most recent value available corresponds to TEC measurements made six months ago. To enable TEC data to be modelled as soon as they are measured, and for future predictions of TEC to be made, the monthly mean sunspot number values were modeled using a smooth curve defined by a summation of sinusoids.

4 The proposed Evolutionary Algorithm

Genetic Programming (GP) is an Evolutionary Computation (EC) technique that evolves populations of computer programs as solutions to problems. It is a general purpose evolutionary search technique that can be applied in both regression and classification problems. In contrast with linear and non-linear regression, the technique does not presuppose as functional form as it generally proceeds for a definition of lower-level building blocks (the function set). Therefore Koza [5] speaks of symbolic regression or system identification as the object of search is both the functional form and the optimal coefficients.

The term evolutionary algorithm describes a class of stochastic search procedure inspired by principles of natural genetics and survival of the fittest. They operate through a simulated evolution process on a population of solution structures that represent candidate solutions in the search space. Evolution occurs through (1) a selection mechanism that implements a survival of the fittest strategy, and (2) diversification of the selected solutions to produce offspring for the next generation.

In GP, programs are usually expressed using hierarchical representations taking the form of syntax-trees. It is common to evolve programs into a constrained, and often problem-specific user-defined language. The variables and constants in the program are leaves in the tree (collectively named as terminal set), whilst arithmetic operators are internal nodes (collectively named as function set). It is common in the GP literature to represent expressions in the prefix notation similar to that used in LISP or Scheme. For example, `x+3*x` becomes `(+ x (* 3 y))`.This representation eases the expression-tree data structure formation, and its manipulation during the application of variation operators.

GP finds out how well a program works by running it, and then comparing its behaviour to some ideal. We might be interested, for example, in how well a program predicts a time series or controls an industrial process. This comparison is quantified to give a numeric value called *fitness*. Those programs that do well are chosen to breed, and produce new programs for the new generation. The primary variation operators to perform transitions within the space of computer programs are crossover and mutation. Once a stopping criterion has been met the algorithm terminates the best program is designated as the output of the run.

The most commonly used form of crossover is subtree crossover [5]. Given two parents, subtree crossover randomly (and independently) selects a crossover point (a node) in each parent tree. Then, it creates the offspring by replacing the subtree rooted at the crossover point in a copy of the first parent with a copy of the subtree rooted at the crossover point in the second parent. Copies are used to avoid disrupting the original individuals. This way, if selected multiple times, they can take part in the creation of multiple offspring programs. Note that it is also possible to define a version of crossover that returns two offspring, but this is not commonly used. Often crossover points are not selected with uniform probability. Typical GP primitive sets lead to trees with an average branching factor of at least two, so the majority of the nodes will be leaves. Consequently the uniform selection of crossover points leads to crossover operations frequently exchanging only very small amounts of genetic material (i.e., small subtrees); many crossovers may in fact reduce to simply swapping two leaves. To counter this, Koza [5] suggested the widely used approach of choosing functions 90% of the time and leaves 10% of the time.

The most commonly used form of mutation in GP (which we will call subtree mutation) randomly selects a mutation point in a tree and substitutes the subtree rooted there with a randomly generated subtree. Another common form of mutation is point mutation, which is GP's rough equivalent of the bit-flip mutation used in genetic algorithms. In point mutation, a random node is selected and the primitive stored there is replaced with a different random primitive of the same arity taken from the primitive set. When subtree mutation is applied, this involves the modification of exactly one subtree. Point mutation, on the other hand, is typically applied on a per-node basis. That is, each node is considered in turn and, with a certain probability, it is altered as explained above. This allows multiple nodes to be mutated independently in one application of point mutation.

Like in other evolutionary algorithms, in GP the individuals in the initial population are typically randomly generated. Two dominant methods are the *full* and *grow* and the widely used combination of the two known as *Ramped half-and-half* [5]. In both the *full* and *grow* methods, the initial individuals are generated so that they do not exceed a user-specified maximum depth. The depth of a node is the number of edges that need to be traversed to reach the node starting from the tree's root node (the depth of the tree is the depth of its deepest leaf). The *full* method generates full tree-structures where all the leaves are at the same depth, whereas the *grow* method allows for the creation of trees of more varied sizes and shapes.

The evolutionary algorithm employed is depicted in Figure 4. It is a standard elitist (i.e. the best is always preserved), generational (i.e. populations are arranged in generations, not steady-state), panmictic (i.e. no program mating restrictions) genetic algorithm [8]. The algorithm uses tournament selection with a tournament size of 7. Evolution proceeds for 50 generations, and the population size is set to 1000 individuals. Ramped-half-and-half tree creation with a maximum depth of 6 is used to perform a random sampling of rules during run initialisation. Throughout evolution, expression-trees are allowed to grow up to depth of 12. The evolutionary search employs a mixture of mutation-based variation operators, where subtree mutation is combined with point-mutation; a probability governing the application of each, set to 0.6 in favour of sub-tree mutation. Neither recombination, nor reproduction was used. The primitive language consisted of the basic arithmetic operators (+, -, *, /) serving as the function set, whereas the terminal set consisted of the six independent variables.

Algorithm 1 Genetic Programming

1: Randomly create an initial population of programs from the available function and terminal sets using Ramped half-and-half tree-creation method.
2: **repeat**
3: Calculate Root Mean Squared Error (RMSE) of each program on the fitness-evaluation data set.
4: **repeat**
5: Randomly select 7 programs from the population.
6: Choose the one with the lowest RMSE or chose the smaller expression tree in case of two of more programs with identical lowest RMSE.
7: Apply subtree or point-mutation to the selected individual.
8: **until** new population has been formed
9: Use the newly created population to define a Pareto-front of individuals in terms of lowest RMSE on fitness-evaluation set (objective 1), and smallest tree-size (objective 2).
10: Evaluate the individuals of the Pareto-front on the validation data-set, and mark the current elitist as the program with the lower RMSE.
11: **until** the number of generations have elapsed.
12: **return** the elitist individual, and test it using the data in the 10^{th} fold.

Figure 4. Genetic Programming approach

5 Experimental Results and Discussion

The primary goal of our experimental studies is to investigate the performance of our GP-based approach in designing a prediction model for the TEC over Cyprus with a good approximation to the measured values, compared to the existing Neural Network based model. The NN used in [6] had a fully connected two-layer structure, with 5 input, 10 hidden and 1 output neurons. Both their hidden and output neurons consisted of hyperbolic tangent sigmoid activation functions. The number of hidden neurons was determined by trial and error. The training algorithm used was the Levenberg-Marquardt back propagation algorithm.

The data-set was segmented in 10 continuous folds similarly to the case of NNs [6]. In each cross-validation cycle, 9 folds are used as the training set, whereas the evolved model is tested on the remaining 10^{th} fold. The training set is further randomly divided into two data-sets (with no overlapping): the fitness evaluation data-set, with 67% of the training data, and the validation data-set with the remaining 33%. The fitness measure consists of minimising the RMSE on the fitness evaluation data-set. At each generation, a two-objective sort is conducted in order to extract a set of non-dominated individuals [7] (Pareto front) with regards to the lowest fitness evaluation data-set RMSE, and the smallest model complexity in terms of expression-tree size, as measured by the number of tree-nodes [9]. The rationale behind this is to create a selection pressure towards simpler prediction models that have the potential to generalise better. These non-dominated individuals are then evaluated on the validation data-set, with the best-of-generation prediction model selected as the one of these with the smallest RMSE. During tournament selection based on the fitness evaluation data-set performance, we used the model complexity as a second point of comparison in cases of identical error rates. The two approaches were coded in Java and run on an Intel® Pentium 4 3.2 GHz Windows XP server with 1.5 GB RAM. We performed 50 independent evolutionary runs for each test fold, in order to account for the stochastic nature of the adaptive search algorithm, and obtain statistically meaningful results.

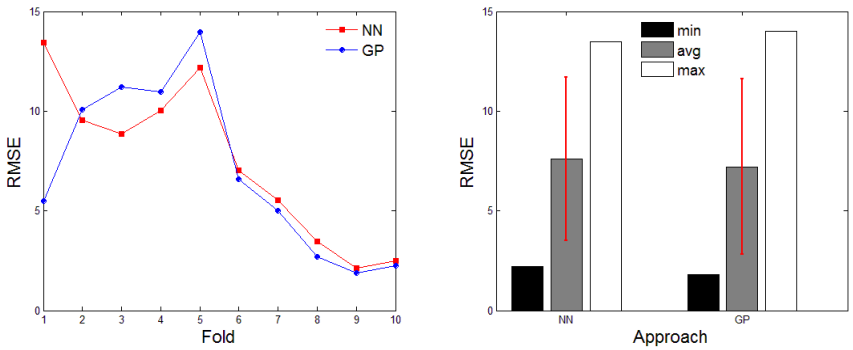


Figure 5. GP versus NN in terms of Root Mean Square Error (RMSE) per fold and average Root Mean Square Error (RMSE).

Figure 5 shows a comparison between the GP and the NN approaches in terms of min RMSE per fold and average RMSE, respectively. The results in Figure 5 show that the GP model provides a lower RMSE compared to the NN-based model in 6 out of 10 folds, giving a better prediction and consequently a better approximation in around 60% of the TEC measured values. However, the NN-based model outperforms the GP-model in folds 2, 3, 4 and 5. The average RMSE obtained by the GP approach is around 4% less than the average RMSE obtained by the NN approach. The two approaches have a similar standard deviation of around 4 RMSE. Some examples of measured and predicted TEC values are given in Figure 6. These demonstrate both the good performance of the developed GP and its superiority over the NN model.

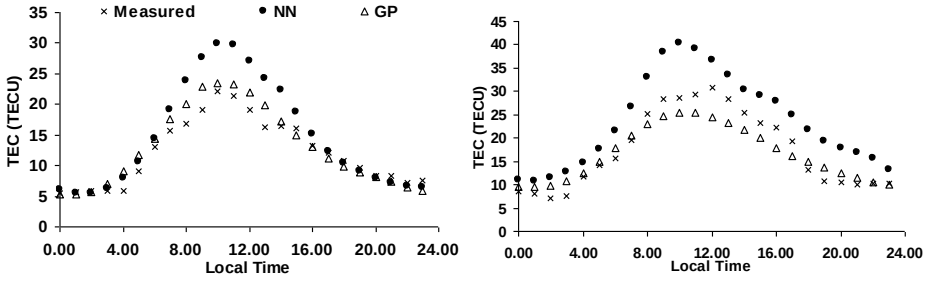


Figure 6. Examples of measured and predicted TEC values.

6 Conclusions and Future Work

In this paper, a Genetic Programming based approach is used to design a prediction model for the Total Electron Content over Cyprus. Particularly, a panmictic, generational, elitist genetic algorithm with an expression-tree representation is used. A prediction model is developed based on a data set obtained during a period of eleven years covering a full sunspot cycle. The GP-model has shown a good approximation of the different time-scales in the variability of the modelled parameter and it has outperformed the existing Neural Network based model. The proposed model can therefore be used in single frequency GPS navigation system receivers to account for the ionospheric imposed error in positioning.

There are a number of avenues for future research. For example, it will be interesting to investigate different genetic operators and primitive languages to further improve the performance of the GP approach. Moreover, the hybridization of the GP with NNs and the design of a more robust approach is also a future possibility.

References

1. J. Goodman.: HF Communications, Science and Technology. Nostrand Reinhold, (1992).
2. N. Maslin: The HF Communications, a Systems Approach, San Francisco (1987).
3. Klobuchar J. A, "Ionospheric Time-Delay Algorithm for Single-Frequency GPS Users," IEEE Trans. on AES, 23 (3), 325-33 1, (1987).
4. C. Reeves, Genetic algorithms, Handbook of Metaheuristics, Kluwer, pp. 65–82, (2003).
5. Koza, J.R., "Genetic Programming: on the programming of computers by means of natural selection", Cambridge, MA, MIT Press, (1992).
6. H. Haralambous, P. Vrionides, L. Economou and H. Papadopoulos. "A local Total Electron Content Neural Network model over Cyprus." Proceedings of the 4th International Symposium on Communications, Control, and Signal Processing (2010)
7. K. Deb, "Multi-Objective Optimization Using Evolutionary Algorithms". Wiley and Sons, 2002.
8. David Goldberg, "Genetic Algorithms in Search Optimisation and Machine Learning", Addison-Wesley, 1989.
9. Christian Gagne, Marc Schoenauer, Marc Parizeau and Marco Tomassini, "Genetic Programming, Validation Sets, and Parsimony Pressure", Proceedings of the 9th European Conference on Genetic Programming, Springer, 10-12 April, 2006.