CHAPTER 4

GENETIC PROGRAMMING APPROACH

4.1 Method of Analysis

The GP is a powerful algorithm among the family of evalutionary computational methods. The GP is based on the biological evolution of the computer programs resulting in an optimum mathematical model of the system.

“For the conventional genetic programming, the structures undergoing adaptation is a population of individual points from the search space, rather than a single point. Genetic methods differ from most of the other search techniques in that they simultaneously involve a parallel search involving hundreds or thousands of points in the search space.”[23]

As in the biological evolution, computer programs (individuals) whose result best fits the observed values, the succesful ones, can survive as they are, whereas unsuccessful individuals are crossed over or mutated. As can be understood from the statement above, the operations taking place in the GP are reproduction of best fitting individuals, crossing over of individuals with lower fitness and mutation of remaining unsuccessfull individuals.

Each function taking part in the Genetic Programming is named as an individual. The characteristics of an individual is represented as a tree structure which is shown in Fig.4.1 as opposed to the characteristics of human, DNA, which is also shown in Fig. 4.2.

A node of an individual is a joint in a tree structure. A terminal point is the node that the mathematical operation are executed. For ease of use, an arm is named as a tree structure under a node.

For the creation of individuals and generations in the process, a certain set of function to be

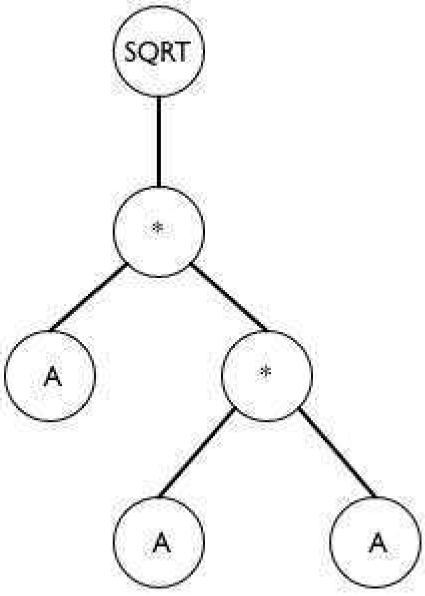

Figure 4.1: A sample tree structure of an individual

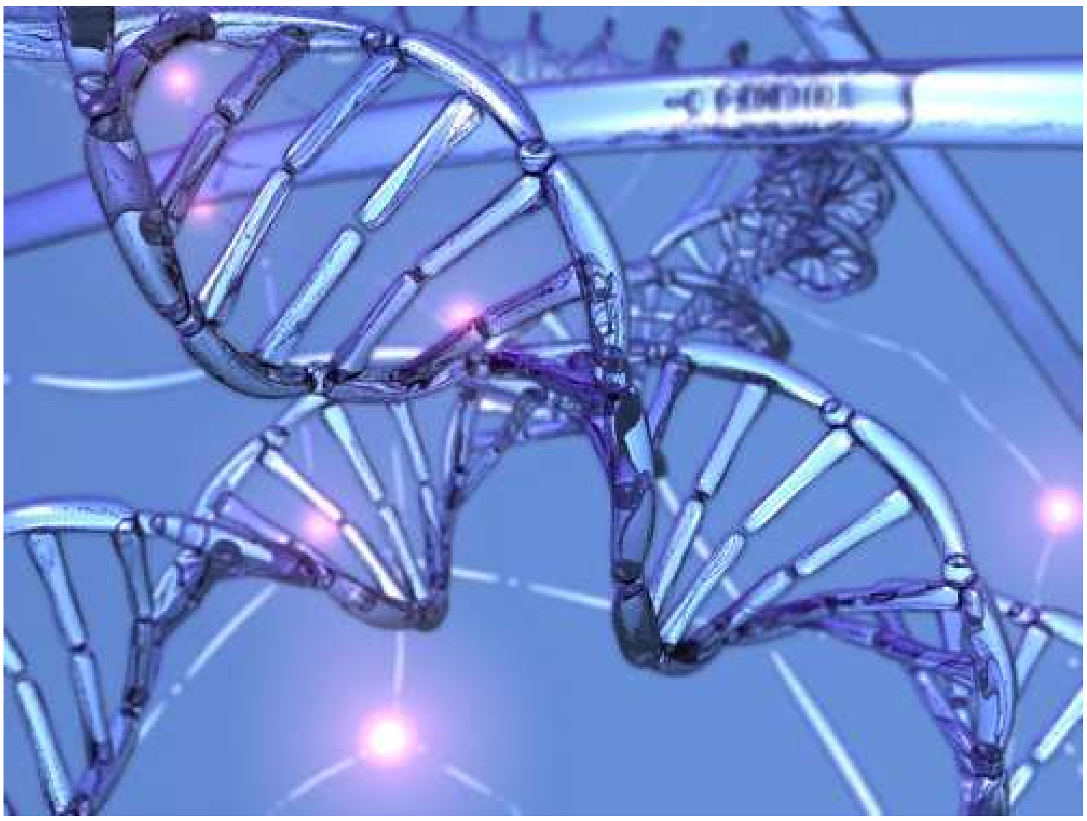

Figure 4.2: Deoxyribonucleic Acid (DNA)

used at terminal points should be created. Depending on the problem of interest, the number and type of the functions are determined.

4.1.1 Operators

In order to obtain of new generations, some operators should act on individuals. Analoguous to biology, individuals will reproduce, crossover with each other or can mutate. Certain fractions of generation with high fitness with respect to other individuals are reproduced to next generation to carry the fitness information to the next generation. Some fraction of the generation with less fitness are crossed over to create more successful individuals fitting the requirements. The remaining fraction of the generation, unsuccessful individuals that should not survive anymore, are mutated. Although in biological situation, most of the time, mutation does create individuals with defects; for GP case, it is assumed that there cannot be any individual worse than the un-mutated individual. In addition, for some cases, mutation results in a jump in the evolution of individual, increasing overall fitness of generation.

From a computer program perspective, the operators can be defined as follows:

- Reproduction
  An individual with a high fitness is passed to next generation without any change.
- Crossover
  Crossover is applied on an individual by simply switching one of its nodes with another node from another individual in the population. (Fig.4.3, Fig.4.4)
- Mutation
  Mutation does not involve any pairs, but it affects an individual in the population. It can replace a whole node in the selected individual, or it can replace just the node's information.

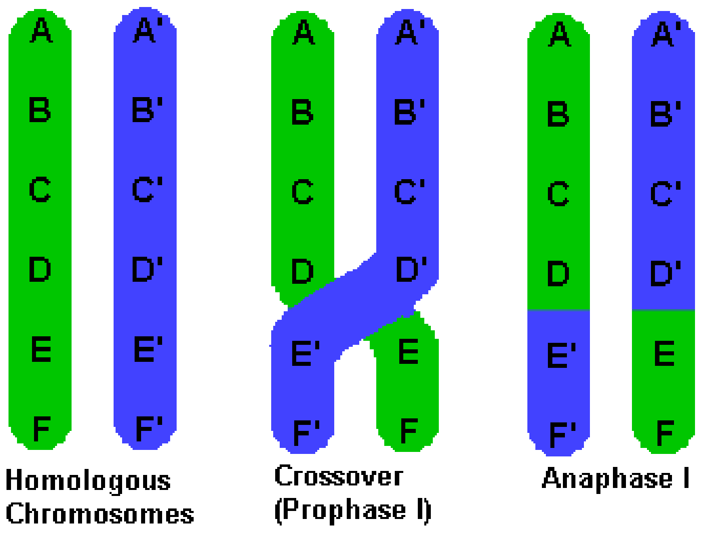

Figure 4.3: The crossing-over process for human DNA

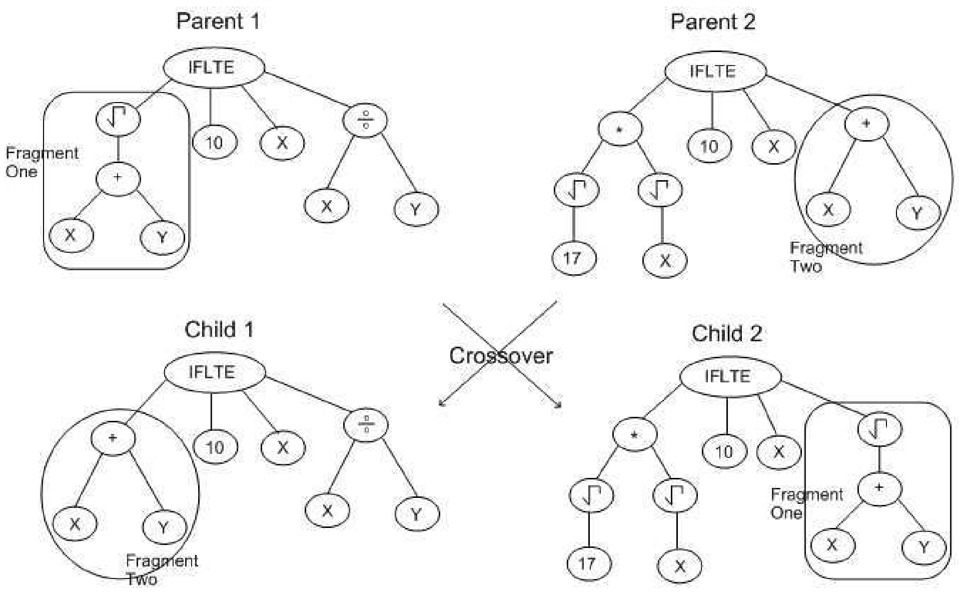

Figure 4.4: A sample crossing over operation

4.1.2 Function Sets

As described above, the terminal points are the points where the mathematical functions are executed. For every individual of the generation of interest, there should be some certain number of terminal points to exist. The choice of a function set for the GP is mostly dependent on the structure of the problem to be solved. For example, if one is given a simple dataset of a polynomial, intuatively, the system can be solved by simple algebraic functions, F={+, -, *, /}. For more complex input/output relations, some other functions like trigonometric, power or exponential functions should be employed. However, unless a small tree of individual is assumed, exponential functions and power functions are useless. Such functions cause *floating point over-flow* error, since the depth of tree gradually increases the numbers to evaluate.

In addition to the simple algebric and trigonometric functions, F={+, -, *, /, sin, cos}, some specially introduced functions can be added to the function set.

4.1.3 Creating Initial Generation

There are various techniques of creating an initial generation. These are “full”, “grow” and “ramped-half-and-half” generation techniques. In full generation technique, all arms of the trees of the individuals are extended to a certain depth specified by the user. For the grow generation technique, all the arms of the trees of the individuals are not to be extended to a certain specified depth specified by the user, but at least one arm of the tree is ought to extend to the maximum depth. The ramped-half-and-half method is just a combination of the previously described methods.[23]

4.1.4 Creating Initial Individuals

The initial individuals are constructed using a random number generator. Depending on the inputs, maximum depth for new individuals, function set and generation technique, functions were placed at the nodes of tree structure without exceeding the specified maximum depth for new individuals.

4.1.5 Generations and Error Analysis

At each generation the individuals were sorted in accordance with their fitness to observed values. Fitness, for this work, was defined as the normalized relative error which is:

$$\text{Norm. Err.} = \frac{1}{N}\sum_{1}^{N}\frac{||\text{Observed Value - Computed Value}||}{\text{Observed Value}} \tag{4.1}$$

where N was the number of data used in testing.

The best fitting individuals were reproduced, and less fitting ones are crossed-over. The remaining worst individuals were mutated. This procedure was repeated until the convergence criterion, which corresponds to a normalized relative error of 2.5%, has been met or the maximum number of generation has been achieved.

4.1.6 Genetic Programming for f_0F2 Values, GETY-IYON

Since response of the Ionospheric variability could not have been shown easily for the IMF B_y events, a different method, Genetic Programming (GP) approach was employed. Shortly, the GP was employed to model the probable effects of polarity reversals of IMF B_y and B_z components on the f_0F2 variability. Thus, event definitions have been modified such that only polarity reversals of IMF B_z and IMF B_y were taken to be seperate and independent events. Depending on these events, GP was used and four independent models were constructed. In order to characterize the eventless periods, one more model was constructed.

In order to have maximum changeability while applying GP, a Genetic Programming code was written in GNU OCTAVE, which is an open source project that can replace the MATLAB and the name of the code was given Genetic Programming by Tolga Yapıcı, GETY.

4.1.6.1 Construction of Models

The input parameters of constructing models were the maximum number of population, size of population, fractions of reproduction, crossover and mutation. Best values for these parameters were obtained by trial-and-error method. The maximum number of population and size of population were kept constant for the construction of all models and the values were:

- Maximum number of populations: 200
- Size of population: 200
- Fraction of Reproduction : 20%
- Fraction of Crossover: 70%
- Fraction of Mutation: 10%

However, since the generation of initial individuals relies on random number, the same results cannot be obtained.

From the classical GP point of view, the mutation is applied to the individuals that have the worst fitness values. However, in the code GETY, the mutation definition was changed. In the mutation definition, the best fitting individuals were copied over the worst ones as well as being reproduced and the copied individuals were mutated. By this way, the GETY code did also worked as a simple optimization tool.

The input variables for constructing the models were the three consecutive values of IMF B_y, IMF B_z and f_0F2. In order to carry the polarity change information for the next hours, a special technique was applied. The code altered the IMF data in such a way that only the magnitude and the polarity change of the IMF B_y and IMF B_z entered in the GETY code. Except the times of polarity reversal change, the IMF data entering the system was given as 0 (zero). For the times of polarity change, the magnitude of the polarity change were given to the GETY as an input. Then, in order to carry the information of polarity change, linearly decaying values of polarity change were given to GETY. By this way, the GETY could apply the polarity change effect for the post-event times. For the values of f_0F2, Arkhangelsk Vertical Ionosonde data were considered. The data covers a period of years 1973 to 1993. The dataset was seperated into two distinct sets. One of the dataset (1973-1980) was used in order to construct the model, whereas the second dataset (1980-1993) was used in order to test the model. The first data group was also seperated in 5 different groups, which were the inputs of each model and polarity reversals of IMF B_y and B_z.

For the critical value of polarity change, relying on the statistical results obtained in previous part, a polarity change greater than 6 nT/h was selected. Thus, for each polarity reversal case, distinct models were constructed. In total, five models were contructed which are:

- Model 1: Model with all inherent event definitions
- Model 2: Model for S.ward IMF B_z polarity reversals greater than 6 nT/h
- Model 3: Model for N.ward IMF B_z polarity reversals greater than 6 nT/h
- Model 4: Model for E.ward IMF B_y polarity reversals greater than 6 nT/h
- Model 5: Model for W.ward IMF B_y polarity reversals greater than 6 nT/h

Also the structure of the model was represented in Fig.4.5

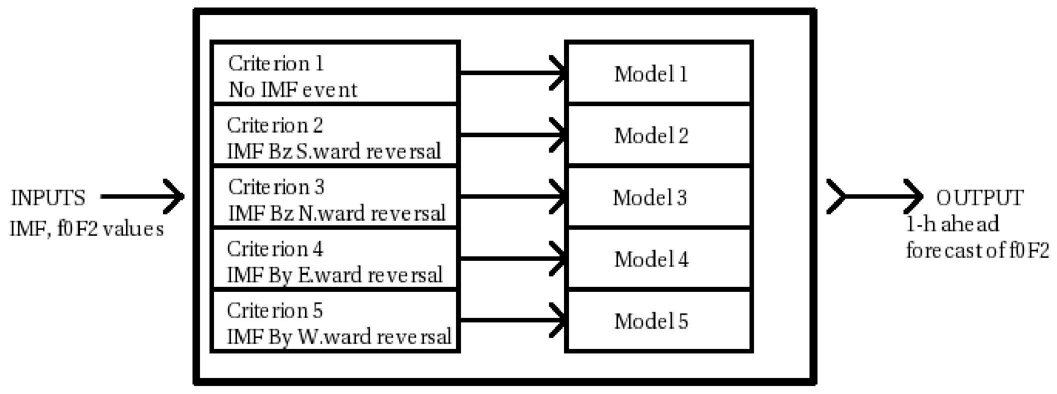

Figure 4.5: Structure of the GETY-IYON

4.2 Results

In Table 4.1, the normalized errors of the seperate models were shown. The high values of error for the Model 1 and Model 2 were the results of the dominance of other events in the interval of interest. However, when one merges 5 models into a single model, then the relative error reduces to 7.3%. This combined model was named as GETY-IYON and the

Fig.4.5 illustrates the working principle of the combined model. In Figure 4.6, observed and computed values of f_0F2 using GETY-IYON and Sunspot Numbers was shown. As can be seen from the figure, the variation of the values of f_0F2 exhibit the dependence on Solar Cycle, which was interpreted from the Sunspot Numbers. As previously described, the data covers two solar cycles which were 1973-1983 solar cycle (used for contructing the models) and 1983-1993 solar cycle(used to test the performance of hte model GETY-IYON). In order to justify the results, results of two selected years which correspond to solar maximum and solar minimum were also ploted.

Table 4.1: Relative Errors of 5 seperate GETY models

	Model 1	Model 2	Model 3	Model 4	Model 5
Normalized Error (%)	17.3	17	9.3	11.2	10.7

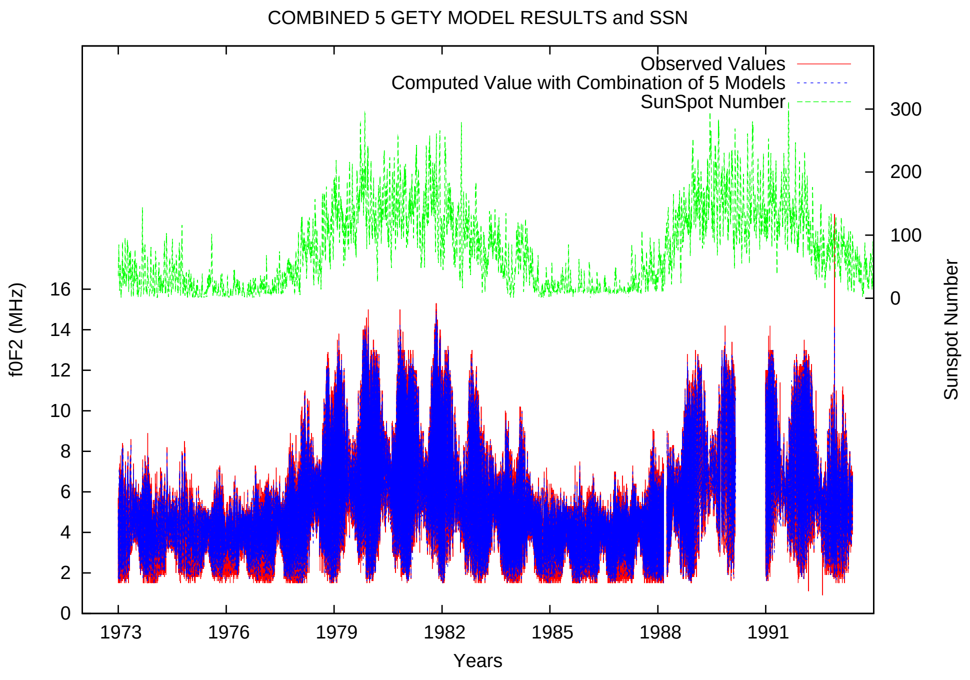

Figure 4.6: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values and the variation of Sun Spot Number

As can be seen from the figures below, the model values well correlated with the observed values at the significance level of 90%. For interpreting the results, next, an extended analysis has been conducted by considering seasonal and monthly data during 1982 and 1987.

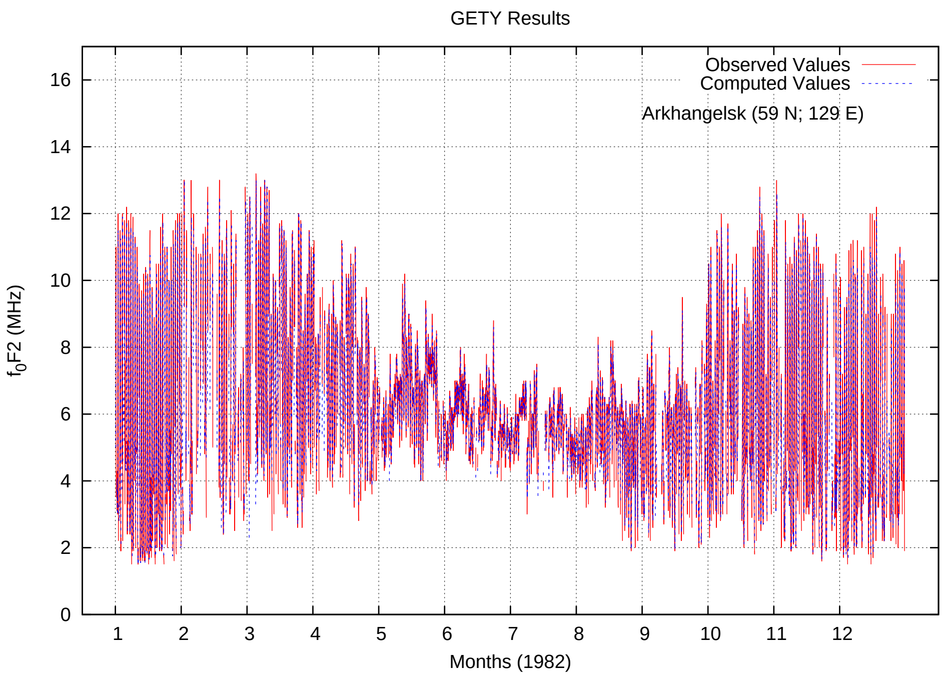

Figure 4.7: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1982

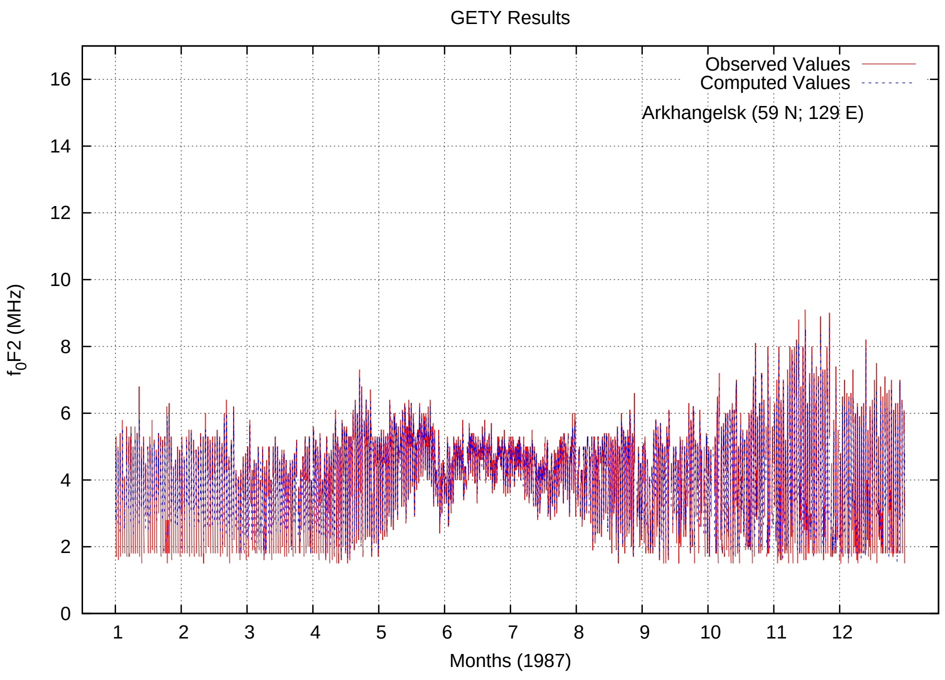

Figure 4.8: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1987

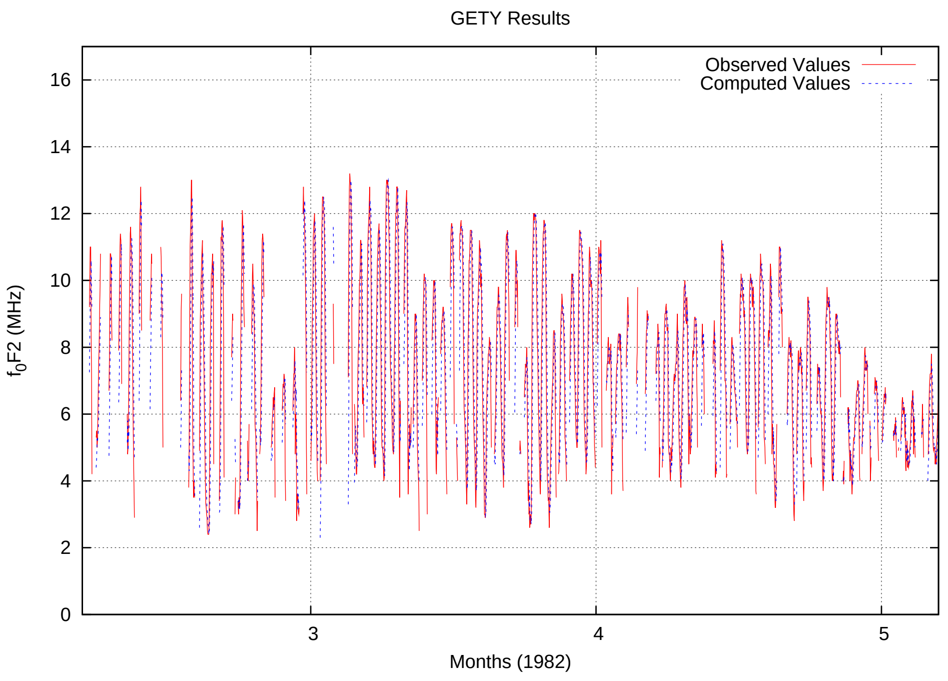

Figure 4.9: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1982 around Spring (Vernal) Equinox

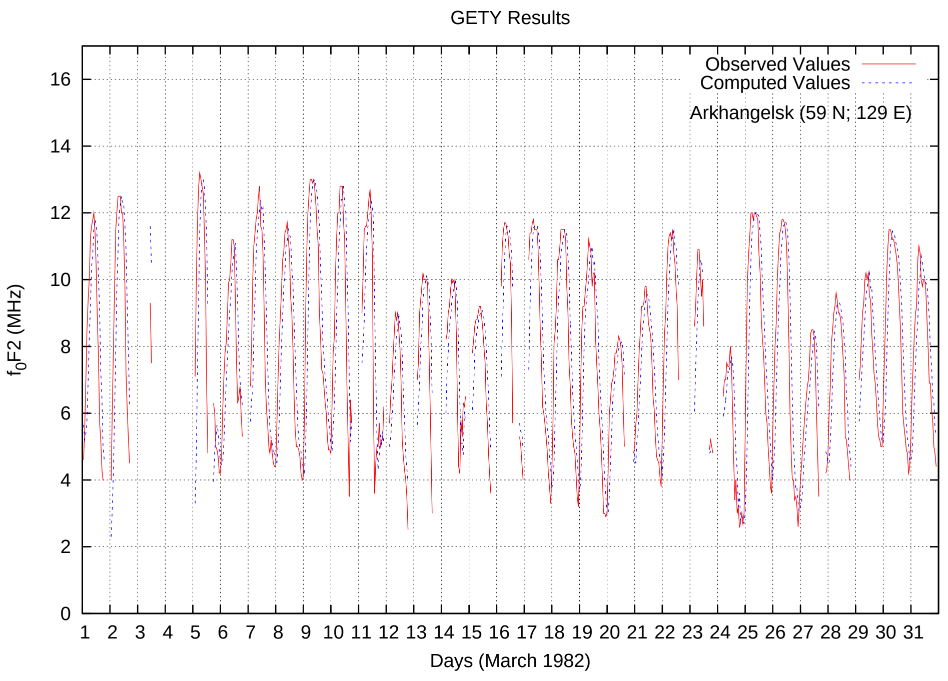

Figure 4.10: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for March 1982 (around Spring Equinox)

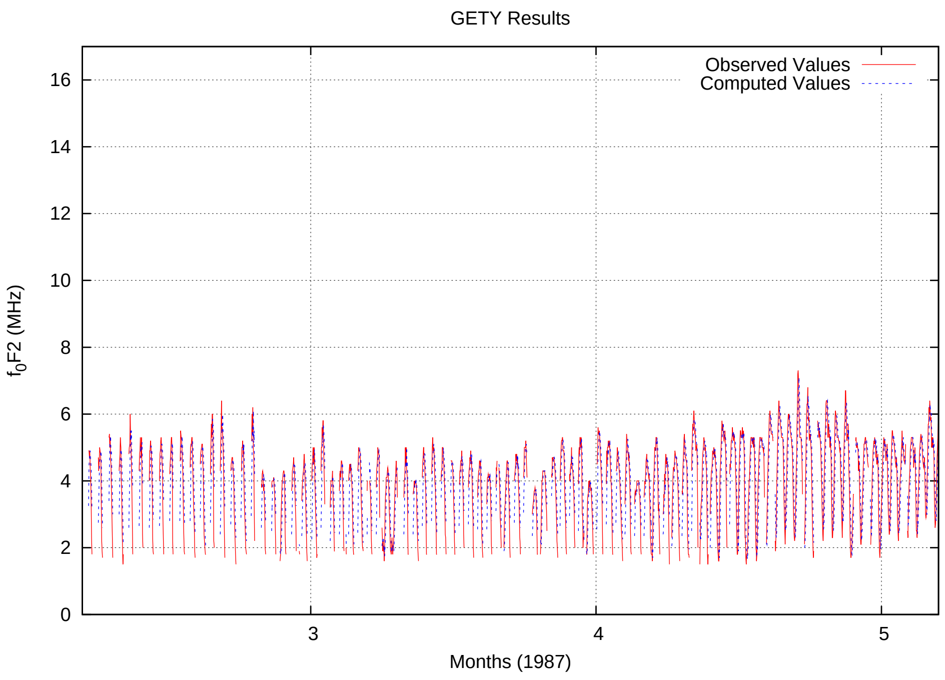

Figure 4.11: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1987 around Spring (Vernal) Equinox

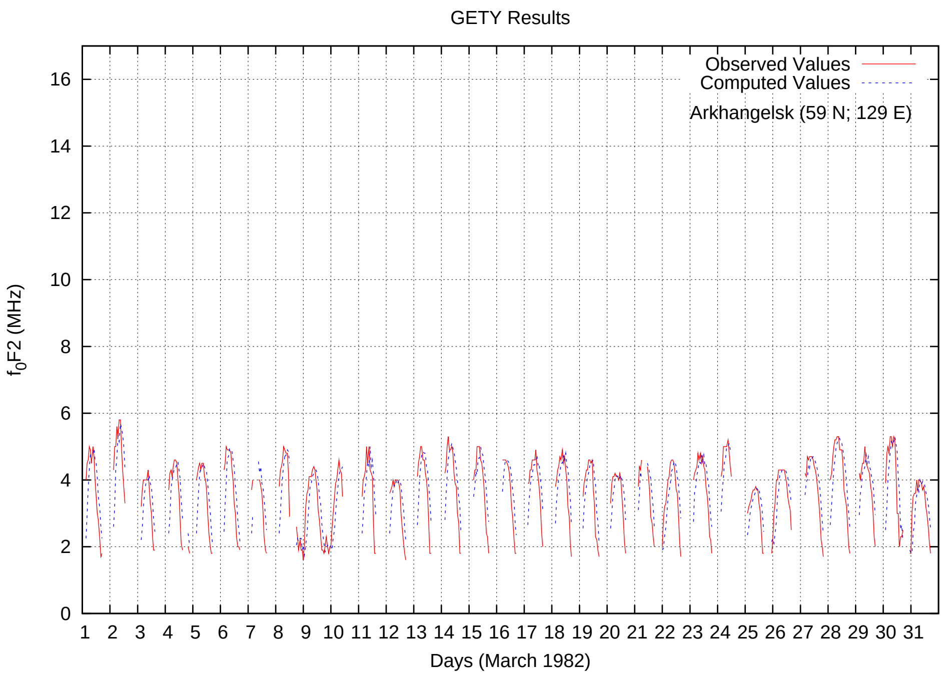

Figure 4.12: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for March 1987 (around Spring Equinox)

The success of the combined model can be seen easily in Figures 4.13, 4.14, 4.15, 4.16 which were plots showing the computed and observed values of f_0F2 during Summer Solstice. The success of the model was high during summer, since the continuous data of f_0F2 during summer and lack of data of f_0F2 during winter impose to GP to create models having behaviours of summer. However, if more data was supplied to the model, then it would have been more successful for the whole year. The performance of the model, GETY-IYON, during Fall and Winter was also as high as performance during summer, but not higher than the performance of summer. The results of fall and winter were shown in Figures 4.17, 4.18, 4.19, 4.20, 4.21, 4.22.

The results plotted monthly showed that the model, GETY-IYON could also be used filling the data gaps.

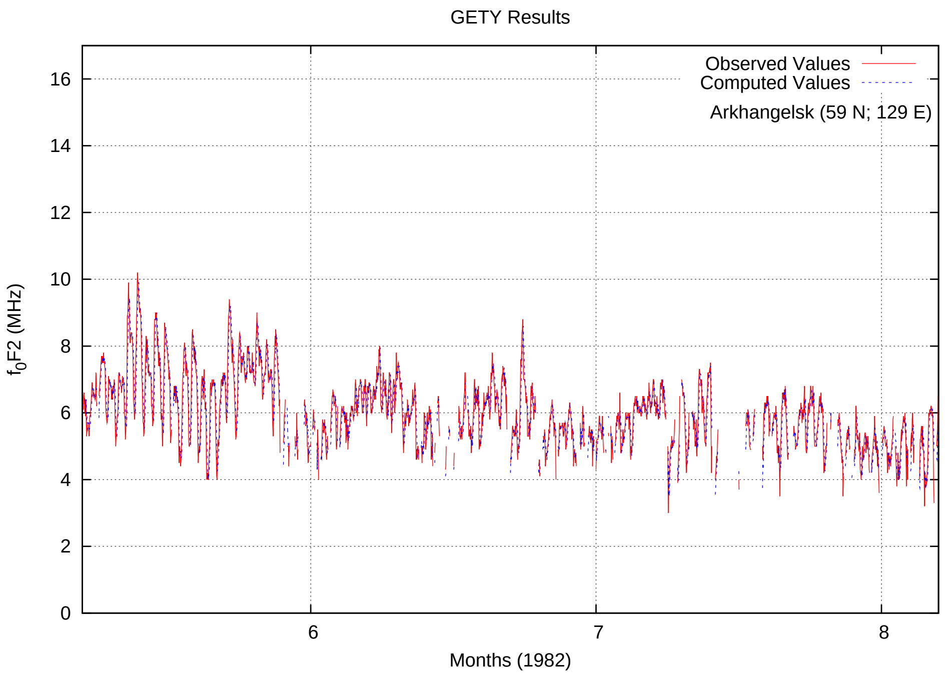

Figure 4.13: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1982 around Summer Solstice

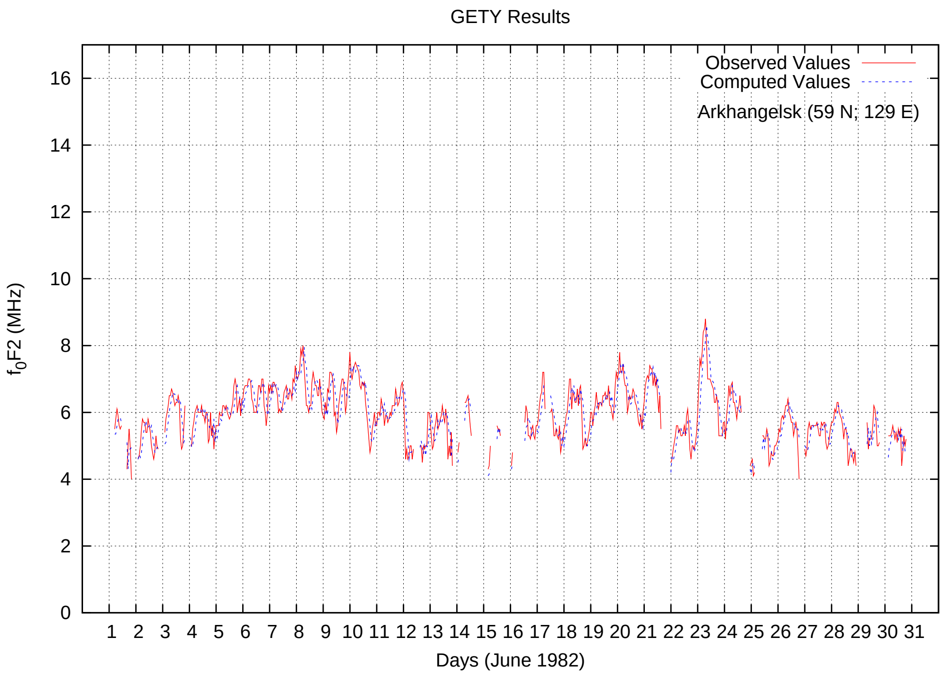

Figure 4.14: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for June 1982 (around Summer Solstice)

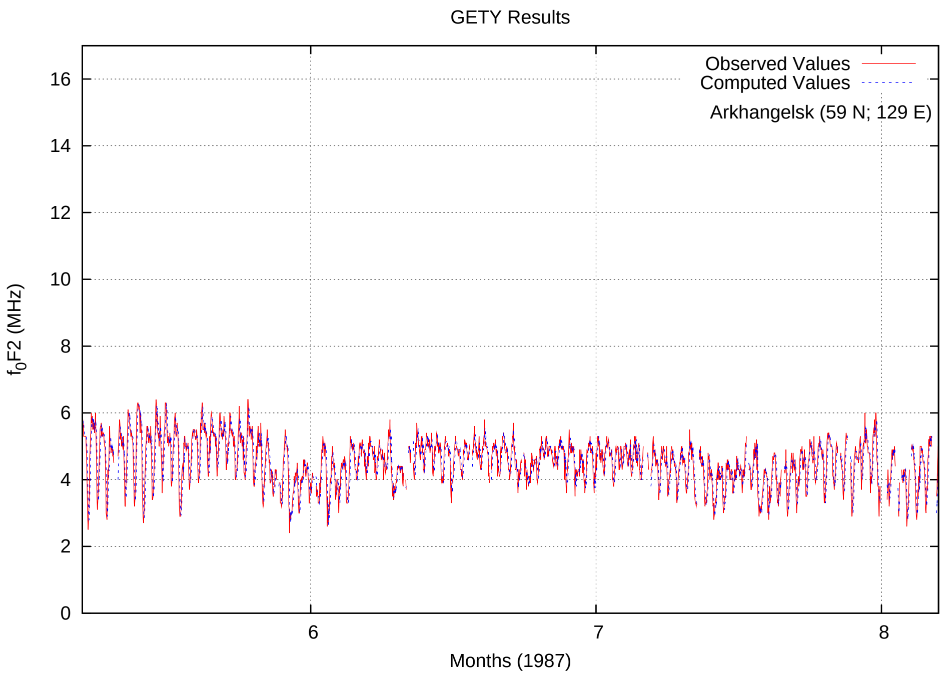

Figure 4.15: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1987 around Summer Solstice

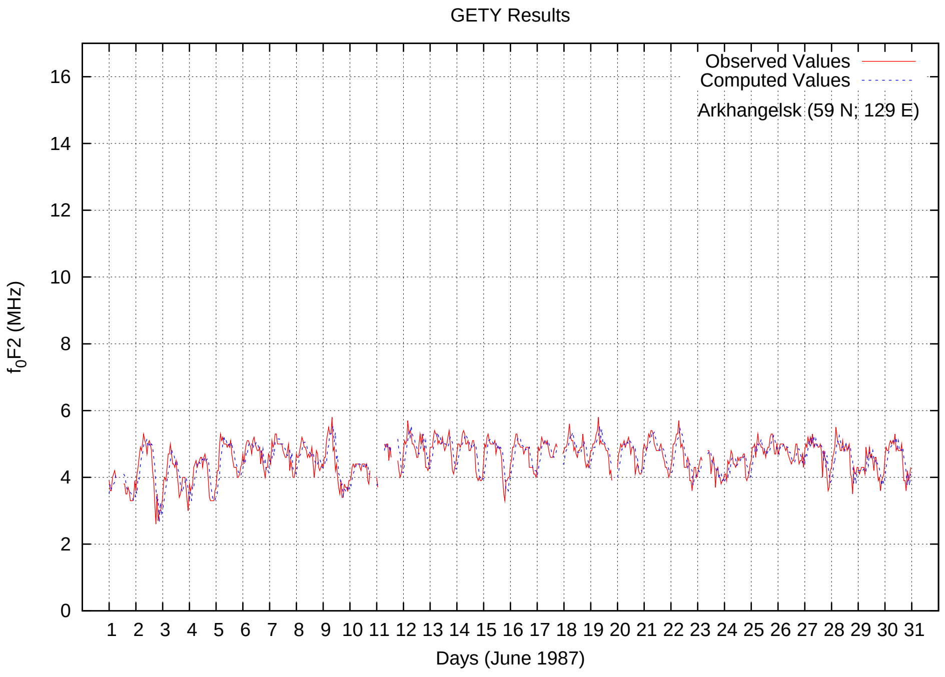

Figure 4.16: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for June 1987 (around Summer Solstice)

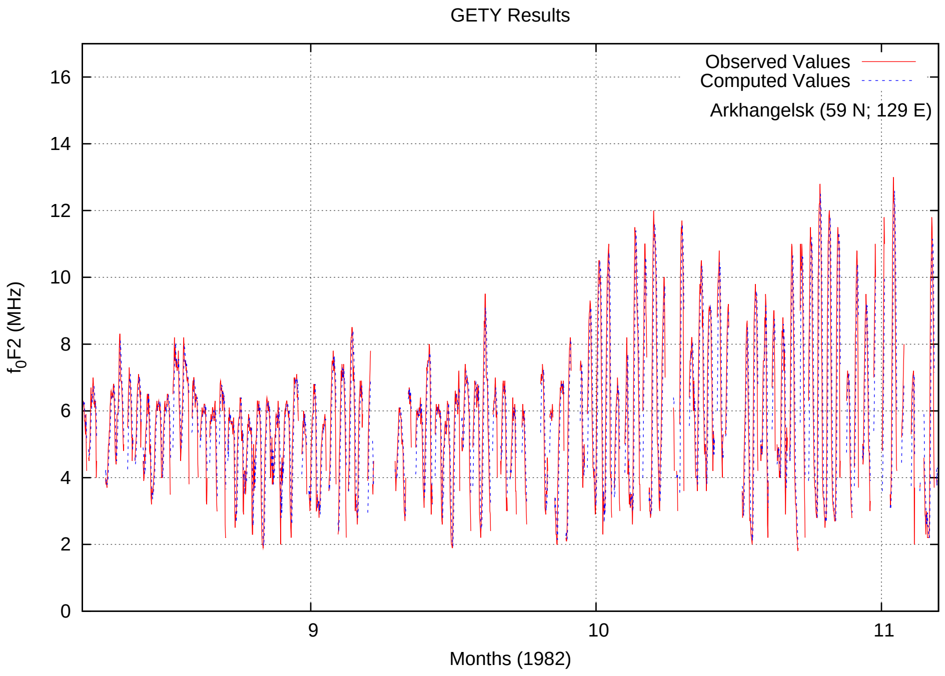

Figure 4.17: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1982 around Fall (Autumnal) Equinox

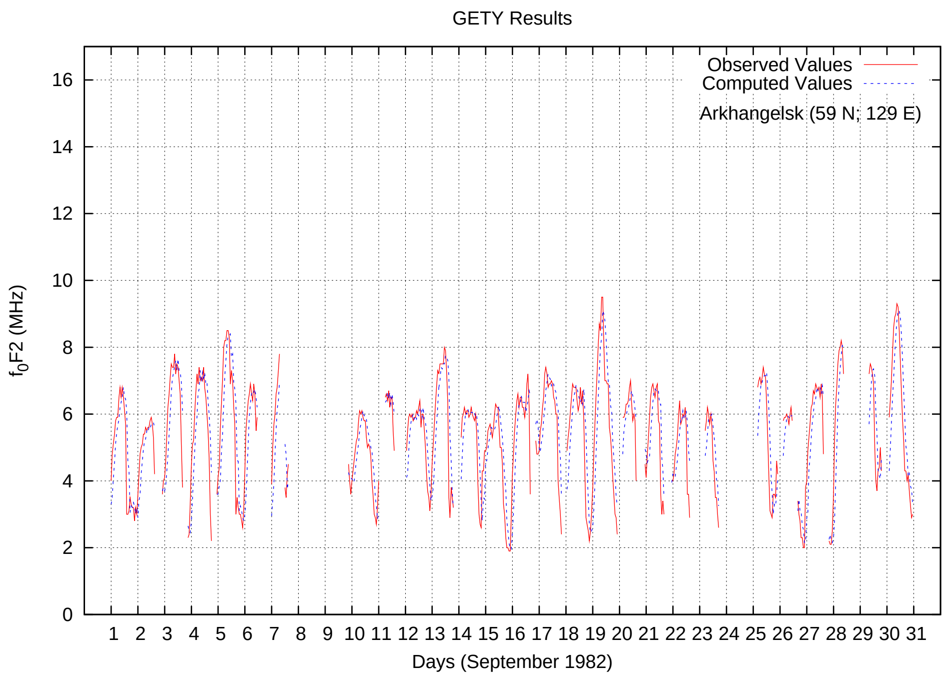

Figure 4.18: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for September 1982 (around Fall (Autumnal) Equinox)

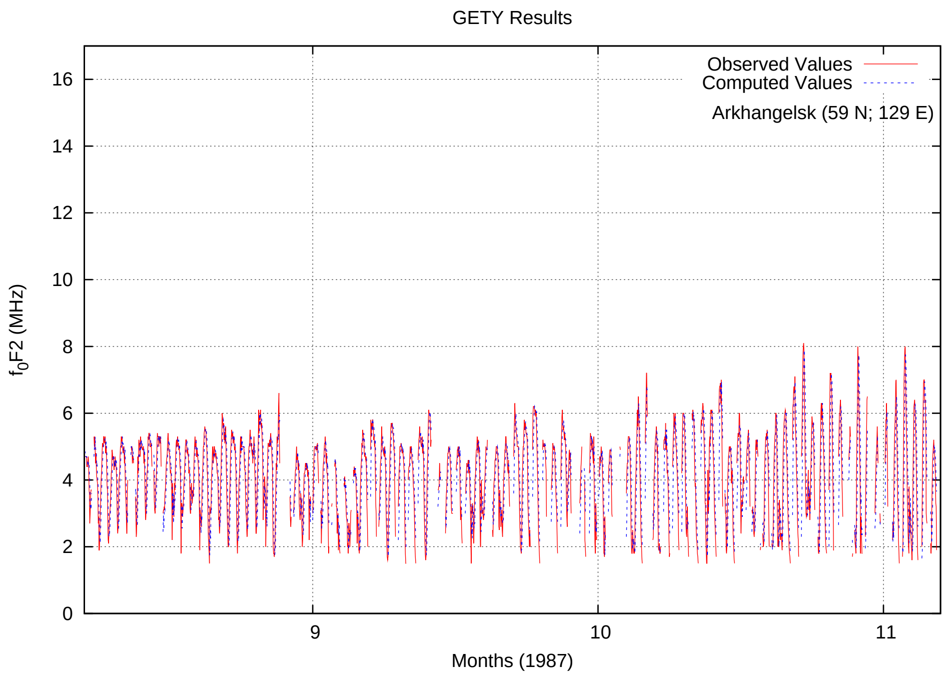

Figure 4.19: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for the year 1987 around Fall (Autumnal) Equinox

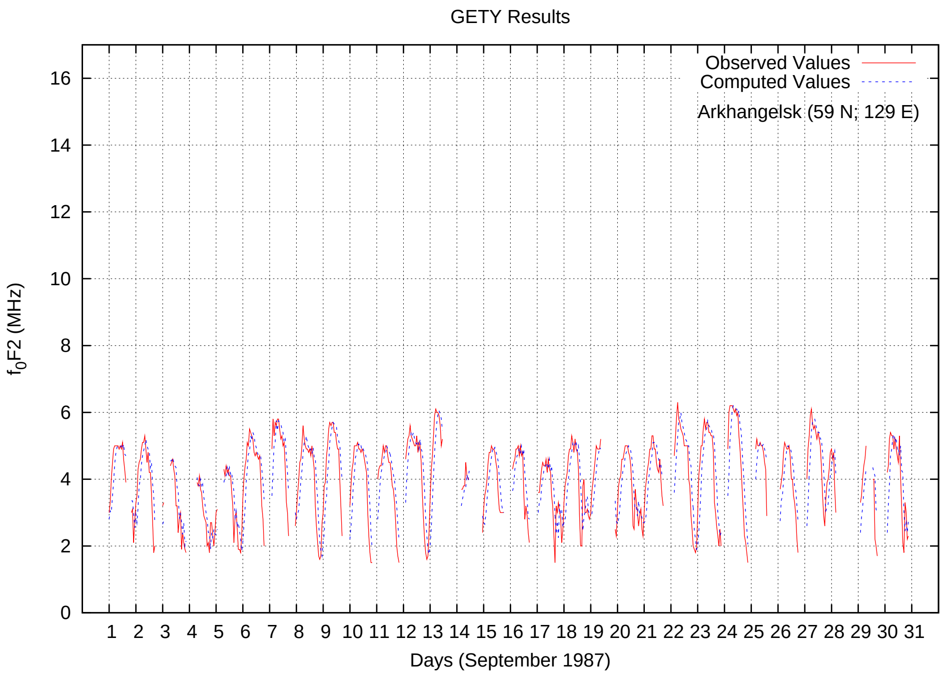

Figure 4.20: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for September 1987 (around Fall (Autumnal) Equinox)

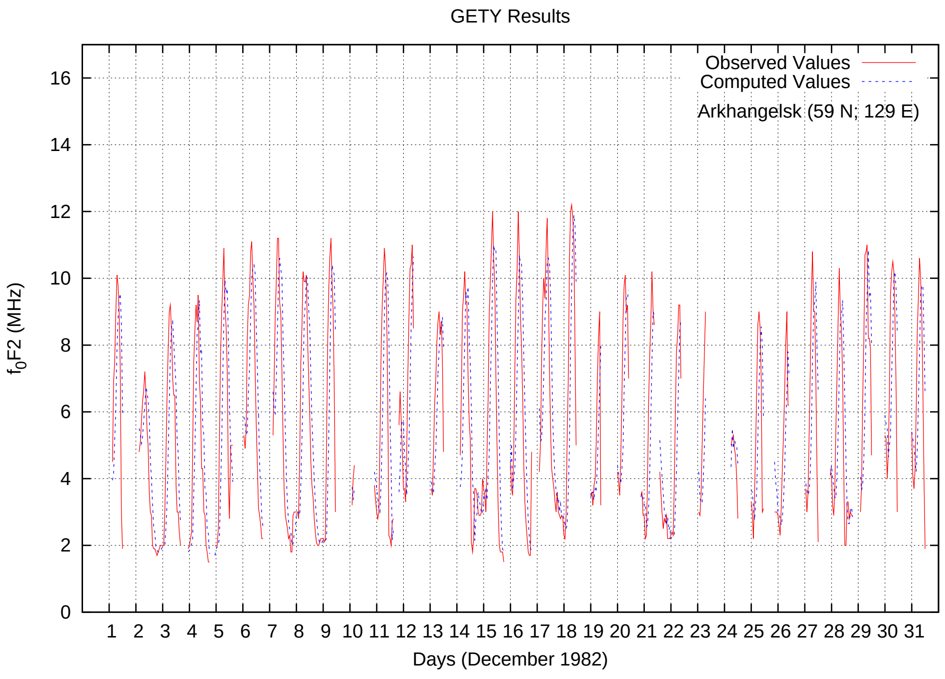

Figure 4.21: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for December 1982 (around Winter Solstice)

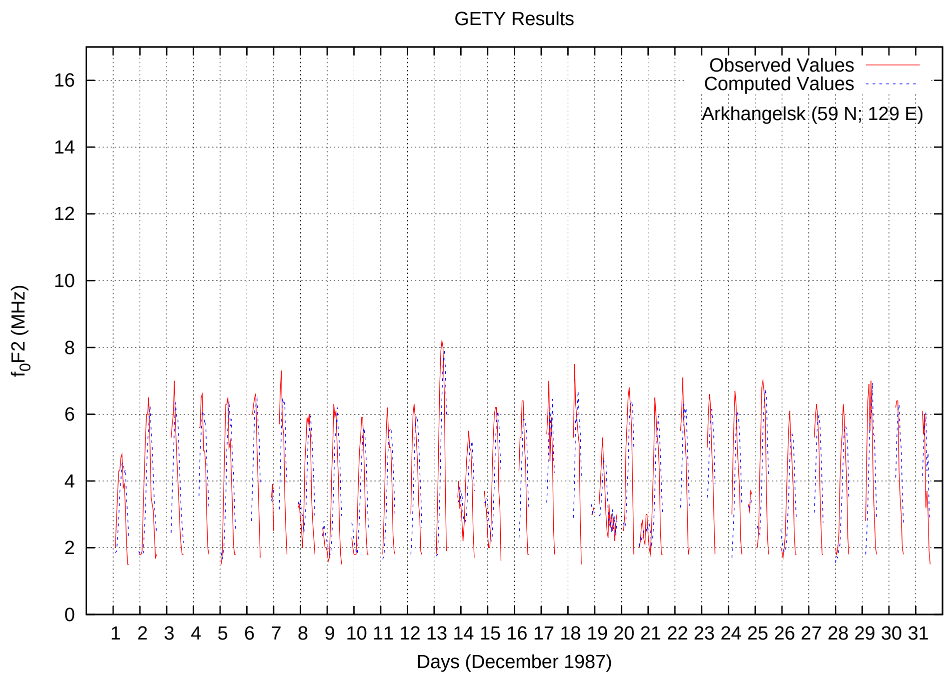

Figure 4.22: Observed Values (in red) and Genetic Programming Results (in blue) of f_0F2 values for December 1987 (around Winter Solstice)