

# EVOLUTION OF SIGNAL PROCESSING ALGORITHMS USING VECTOR BASED GENETIC PROGRAMMING

*K. L. Holladay<sup>1</sup>, K. A. Robbins<sup>2</sup>*

<sup>1</sup>Southwest Research Institute, San Antonio, TX, kholladay@swri.org

<sup>2</sup>University of Texas at San Antonio, San Antonio, TX, krobbins@cs.utsa.edu

## ABSTRACT

This paper demonstrates that FIFTH™, a new vector-based genetic programming (GP) language, can automatically derive very effective signal processing algorithms directly from signal data. Using symbol rate estimation as an example, we compare the performance of a standard algorithm against an evolved algorithm. The evolved algorithm uses a novel approach in developing a symbol transition feature vector and achieves an impressive 97.7% overall accuracy in the defined problem domain, far exceeding the performance of the standard algorithm. These results suggest that vector based GP approaches could be useful in developing more expressive features for a large class of signal processing and classification problems.

*Index Terms*— Genetic Programming, Symbol Rate, Feature Extraction, FIFTH

## 1. INTRODUCTION

Although Genetic Programming (GP) research has been ongoing for several years, there are relatively few applications in digital signal processing. Perhaps one of the reasons for this is the difficulty of applying classical GP approaches to problems that are better expressed as vector transformations. When dealing with real-world time series data, e.g. [1], researchers often pre-process the data to produce features, then use the individual feature points as the input terminal values for GP.

In this paper, we present our current research<sup>3</sup> on using a vector-based GP language to automatically discover signal processing algorithms directly from large sets of time series data. First we introduce our new GP language, FIFTH™, and explain how its design is ideally suited to signal processing algorithms. Next, we present a case for using symbol rate estimation algorithms as an example problem. Finally, we compare the performance of a common symbol rate algorithm with a GP derived algorithm.<sup>4</sup>

---

<sup>3</sup> This work was sponsored by the Southwest Research Institute Advisory Committee for Research (ACR)

<sup>4</sup> Computational support was provided by NIH Research Centers in Minority Institutions grant 2G12RR1364-06A1 and by the UT System San Antonio Computational Biology Initiative

## 3. THE FIFTH GP LANGUAGE

Genetic Programming is a randomized search strategy that searches the space of executable programs rather than the data space for problem solutions. While GP research has produced “human-competitive” solutions in certain arenas [2], it is not often applied directly to time series data. One reason for the lack of progress is the inability of most GP languages to handle vectors as a native data type. Without native vectors, simple operations such as multiplying a scalar and vector require complicated loop structures that are easily destroyed during the evolutionary process. Also, few GP languages include advanced signal processing functions such as the Fourier transform, which are unlikely to be evolved but which are very likely to be needed for a signal processing application.

We recently introduced a new GP language called FIFTH [3] that addresses these deficiencies. Patterned after FORTH, FIFTH is a stack-based language in which a single stack container can hold a scalar, vector or matrix. Containers facilitate direct manipulation of vectors without requiring complicated constructs, resulting in very compact program representations. FIFTH programs are easy to manipulate while guaranteeing syntactically correct programs even through genetic operations.

FIFTH also differs from other GP languages in that it uses a linear representation instead of a tree-based representation. This is more natural for problems in a vector space, which often contain sequential transformation operations. It also introduces a substantial difference in the program search space for mutation and crossover operations.

## 2. SYMBOL RATE ALGORITHMS

Analyzing a digital communication signal without any prior knowledge of its properties typically requires the application of numerous signal processing algorithms to determine values such as modulation type and number of symbol states. For autonomous surveillance applications, determining the order in which to apply the algorithms as well as the algorithm configuration parameters that yield optimal results can be very difficult. Variations in the expected mix of intercepted signals may even require development of new algorithms, which is usually a labor intensive process. An important step early in the analysis process is the estimation of the symbol rate.

Symbol rate (SR) estimation algorithms typify a large class of problems where the raw data is best represented as a vector. For this paper, we presuppose capture of fixed length signals at a fixed sample rate. Each signal is base-band and analytic (complex) in form. The goal is to determine the symbol rate of each unknown signal.

Many different symbol rate estimation algorithms have been developed, analyzed and published [4]. Most SR algorithms are comprised of two computational stages. The first stage develops a feature vector by applying one or more transforms to the digitized signal. The goal is to locate and emphasize symbol transitions. For example, a common algorithm for estimating the symbol rate of a phase shift key (PSK) signal uses the magnitude of the derivative of the signal phase angle [5]. We will refer to this algorithm as DPDT. The feature development stage for DPDT as expressed in FIFTH is as follows ( $x$  is the signal vector):

```
x ANGLE UNWRAP DIFF MAGNITUDE
```

The second stage analyzes the resulting features to produce a single real value representing the symbol rate. The analysis phase often takes a Fourier transform of the first stage vector followed by peak selection to locate the spectral peak corresponding to the symbol rate. The simplest peak selection involves masking off very low frequencies (low order FFT bins) and selecting the largest peak. An implementation of this in FIFTH is ( $F_s$  is the sample rate in samples per second):

```
FFT MAGNITUDE LENGTH 2.0 / SETLENGTH
VRAMP Fs 2 / * 10.0 GT * MAXINDEX
x LENGTH SWAP DROP Fs SWAP / *
```

The first line uses a Fourier transform to develop the periodicity of the feature vector. The second line finds the index of the highest spectral energy above 10 Hz, and the third line uses the sampling frequency to convert the index into a symbol rate leaving a single number on the stack.

The experiment described in this paper used the above code as a common second stage. The GP task was to develop a new first stage feature extraction transformation.

#### 4. EXAMPLE DATA SET

The first step in applying GP is to specify the problem domain and to determine the training set for fitness evaluation of each generation. Table 1 lists common signal, channel, and receiver properties that affect the performance of blind symbol rate estimation algorithms for surveillance systems working in the High Frequency (HF) range of 3-30 MHz.

Using a previously developed algorithm evaluation framework [6], we generated a training set consisting of 3680 signal files including five random symbol variations for each combination of signal properties, a fixed 8000 Hz sample rate, and a fixed signal length of 16384 samples. All FSK signals used continuous phase transitions between symbols. The training set includes complexities that arise in practice but that are often not considered in published stud-

ies, such as pulse shaping and symbol rates that are not integer divisors of the sample rate.

**Tbl. 1.** Properties of the training set signals used for the symbol rate experiments.

Property	Typical values in HF	Values used
Modulation	FSK, MSK, PSK, DPSK, OQPSK, QAM, ASK	FSK, PSK
Pulse shape	None, raised cosine (RC), root RC (RRC), Gaussian	None, RC
Excess bandwidth (rolloff)	Limit: 0.00 to <1.00. Typical: 0.10 to 0.35	0, 0.1, 0.2, 0.35
Mod Index	0.5 to 3	0.1
Symbol rate	Typical: 10 to 2400 symbols per second	13 values in [50, 2400]
Symbol states	2, 4, 8	2, 4, 8
Signal to noise ratio (SNR)	Practical range: 0 to 60 dB	9, 12, 16, Inf

The training set also includes property ranges that are known problems for the DPDT algorithm including Frequency Shift Key (FSK) modulation, slow symbol rates (<100), and low rolloff. Defining a correct answer as one with less than 1% error, the DPDT achieved only 66.8% correct responses against the training set. For PSK only, the domain in which DPDT should work well, it achieved only 78% correct estimations.

#### 5. EVOLVED SYMBOL RATE ALGORITHM

A number of configuration parameters are required to set up the FIFTH GP environment. The example in this paper used a configuration consisting of 4000 programs per generation, minimum program size of 5 tokens, maximum program size of 75 tokens, parent pool size of 3500 programs, exponential fitness ranking with a 0.9 bias, and the terminals and functions shown in Table 2. The genetic operations included probabilities 0.8 for mutation and 0.2 for crossover. Both genetic operations used a linear probability length selection between 1 and 10 program tokens.

**Tbl. 2.** Selection probabilities for tokens used by the random program generator for the SR problem.

P(selection)	Terminals and functions
0.10	x (signal vector)
0.05	1.0 2.0 3.0 4.0 5.0 10.0 -1.0 -2.0
0.10	DUP SWAP ROT OVER
0.10	* + - /
0.65	REAL IMAGINARY ANGLE UNWRAP FFT MAGNITUDE DIFF SQRT RAMP VRAMP MAGSQRD COS SIN FLOOR ROUND WND_HAMMING REVERSE CEILING

The evaluation procedure for each generation used a subset strategy. At each generation a set of 50 test signals was picked at random, and the programs were evaluated on the subset to determine their fitness. Subsetting, which has some of the attractive properties of the bagging and boosting techniques in machine learning [7], speeds up the fitness calculation, prevents over fitting, and avoids premature convergence to local minima.

Figure 1 shows the progression of a GP run by plotting the fitness of the best program in each generation. In generation 24, program 1333 achieved a sufficiently low fitness, and because no better program appeared in the next 10 generations, the run terminated. The resulting best program (hereafter referred to as P1333) is very different from any published algorithm with which the authors are familiar:

```
x SIN FFT DIFF ROUND DIFF FFT x / FLOOR x -
17 x * FLOOR REAL /
```

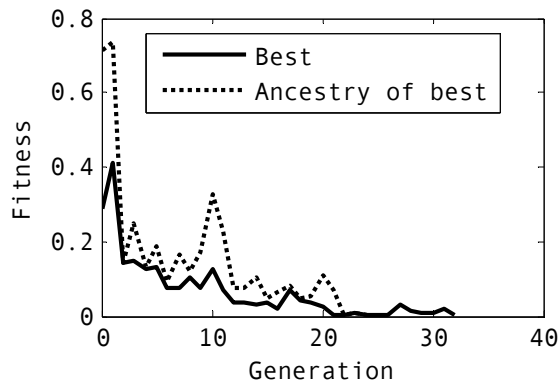


Fig. 1. Fitness progression for an example GP run.

When run against the entire training set, P1333 achieves an impressive 97.7% correct estimation. Table 3 compares the distribution of the correct answers by symbol rate (baud) and modulation type for DPDT and P1333.

To validate the performance of P1333, we ran it against a set of test files that were not used during the evolutionary process. The boundary conditions for the test set were similar to the training set with the following differences. First, two new property values were included: root raised cosine (RRC) pulse shape and phase discontinuous FSK symbol transitions. Second, only five different symbol rates were used. With ten random symbol variations for each combination of property values, the test data set consisted of 6250 signal files.

As seen in Table 4, the performance of both algorithms was similar for both the training set and the test set. When we expanded the FSK signal properties to include other modulation indexes, neither DPDT nor the GP algorithm performed well against the new FSK signals. We are planning additional experiments with an augmented training set to evolve an even more general algorithm.

Tbl. 3. Percent correct performance by symbol rate value against the entire training set.

Baud	DPDT		P1333	
	FSK	PSK	FSK	PSK
50	0.0	45.4	78.3	87.9
75	0.0	59.6	83.3	96.3
96	0.0	65.8	85.0	99.6
100	0.0	72.1	98.3	99.6
125	1.7	72.9	96.7	100.0
200	1.7	75.0	91.7	100.0
300	0.0	75.0	93.3	100.0
600	12.5	100.0	100.0	100.0
800	0.0	75.0	100.0	100.0
1280	0.0	75.0	100.0	100.0
1500	75.0	100.0	100.0	100.0
1800	5.0	100.0	100.0	100.0
2400	NA	100.0	NA	100.0

Tbl. 4. Percent correct performance by symbol rate value against the test set.

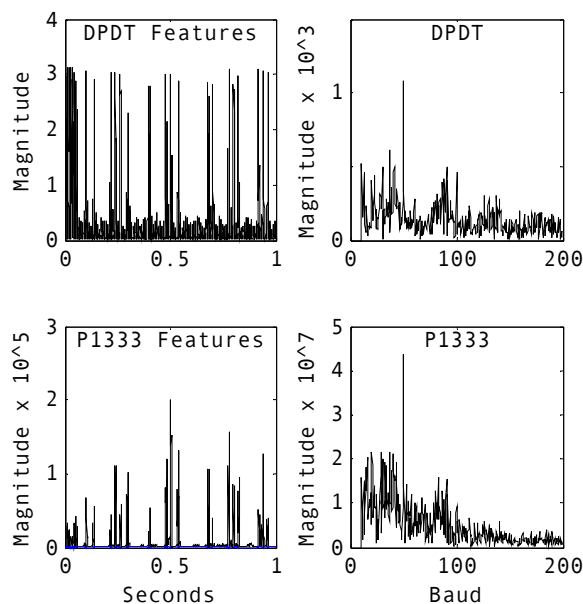
Baud	DPDT		P1333	
	FSK	PSK	FSK	PSK
50	0.3	68.0	74.7	92.8
100	0.0	82.2	99.0	100.0
300	0.0	85.7	89.7	100.0
1280	2.0	85.7	100.0	100.0
2400	NA	100.0	NA	100.0

## 6. P1333 ALGORITHM ANALYSIS

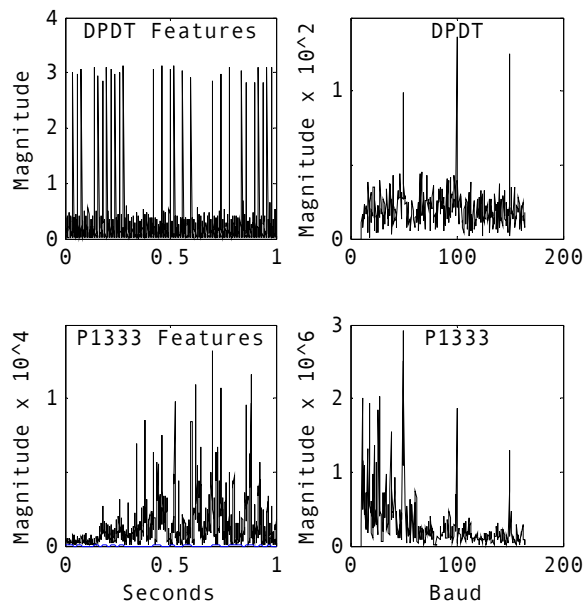
Insight into the operation of P1333 can be gained by comparing its intermediate results with those of DPDT. The two intermediate points of interest are the stage 1 feature vector and the stage 2 vector just after the Fourier transform.

Figure 2 shows the feature vectors and stage 2 vectors for a signal that both algorithms estimated correctly. In the feature plots on the left, the symbol transitions are clearly seen as large spikes, but the spikes in P1333 are several orders of magnitude greater than for DPDT. The plots to the right show the stage 2 vectors with the x axis scaled to represent the symbol rate. The largest peak for both algorithms is located at 50 symbols per second, the correct rate. However, the P1333 peak is again several orders of magnitude larger (4E7 versus 1E3).

Figure 3 shows the equivalent analysis for a similar signal in which DPDT was incorrect and P1333 was correct. DPDT again shows strong peaks at each symbol transition, but the peaks do not line up as well with the P1333 peaks. DPDT exhibits a harmonic peak at 100 baud that is larger than the 50 baud peak. Even though the harmonic is present in P1333, it is lower in magnitude than the 50 baud dominant peak. After examining multiple signals in this way, we conclude that P1333 produces fewer and smaller harmonic peaks than does DPDT.



**Fig. 2.** Feature vector and FFT vector for DPDT and P1333, both correct



**Fig 3.** Feature vector and FFT vector for P1333 correct, DPDT incorrect

## 6. CONCLUSION

The results of this experiment clearly demonstrate that the genetically evolved P1333 algorithm performs significantly better than the commonly used DPDT algorithm over the selected range of signal properties. Based on this and other experiments, we conclude that our FIFTH genetic pro-

gramming environment is a viable technique to address real-world vector space problems. The production of P1333 as a viable algorithm was not a fluke. Subsequent runs produced many potentially useful algorithms for the SR problem. We have also done some promising preliminary experiments to generate full SR algorithms (no pre-defined second stage).

The key to FIFTH's success appears to be its native handling of vectors and its ability to include domain specific primitives (in this case, operators such as FFT and windowing functions) as indivisible operators. We also found that the closure properties of all operators had to be carefully considered. For example, when two vectors of unequal size (or a vector and a matrix) are added --- what should the footprint of the result be? Decisions about these details made a strong impact on the results that were generated.

This work is highly significant because it demonstrates the utility of the GP approach using FIFTH to produce "human competitive" vector algorithms for feature extraction. Feature extraction is the first stage in many problem domains where large volumes of data are naturally expressed as vectors. Without native vectors, it is difficult to formulate these problems so that GP analysis can directly process the raw data. We believe that our results with FIFTH indicate that a large class of these problems may now be more easily addressed using GP.

## 7. REFERENCES

- [1] H. Guo, L. B. Jack, and A. K. Nandi, "Automated feature extraction using genetic programming for bearing condition monitoring," presented at IEEE Workshop on Machine Learning for Signal Processing, 2004.
- [2] J. R. Koza, M. A. Keane, M. J. Streeter, et al., *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*: Kluwer Academic Publishers, 2003.
- [3] K. Holladay, K. Robbins, and J. v. Ronne, "FIFTH™: A stack based GP language for vector processing," presented at EuroGP 2007, Valencia, Spain, 2007.
- [4] K. Holladay and K. Robbins, "Experimental analysis of wavelet transforms for estimating PSK symbol rate," presented at IASTED International Conference on Communication Systems and Applications, Banff, Canada, 2004.
- [5] R. J. Mammone, R. J. Rothaker, and C. I. Podilchuk, "Estimation of carrier frequency, modulation type, and bit rate of an unknown modulated signal," presented at IEEE International Conference on Communications, 1987.
- [6] K. Holladay and K. Robbins, "A framework for automatic large-scale testing and characterization of signal processing algorithms," presented at Military Communications (MILCOM), Monterey, CA, 2004.
- [7] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY: Springer-Verlag, 2001.