Evolutionary Spatial Auto-Correlation for Assessing Earthquake Liquefaction Potential using Parallel Linear Genetic Programming

Aaron Scoble*, Will Browne*, Bill Stephenson†, Zane Bruce†, Mengjie Zhang*
*Victoria University of Wellington, Wellington, New Zealand
{aaron.scoble, will.browne, mengjie.zhang}@ecs.vuw.ac.nz
†GNS Science, Wellington, New Zealand
{bill.stephenson, z.bruce}@gns.cri.nz

***Abstract*—The assessment of sites for liquefaction potential in earthquakes currently relies on the estimation of soil layer models which is laborious and standard regression techniques ineffectual. Although Parallel Linear Genetic Programming (PLGP) has proven to be an effective method for classification tasks it has not yet been applied to regression problems. This paper redefines a time-consuming, operator intensive process as an Evolutionary Computation (EC) regression task and designs a PLGP system that can produce candidate solutions for an operator to review. This paper introduces Evolutionary Spatial Auto-Correlation (ESPAC) which is an EC technique that uses a similar structure to PLGP programs to represent some layer models and evolve them using error matching against the target curve as a fitness function. The project achieves its goal of providing a working proof-of-concept with resultant curve matching being improved over that of a domain expert on four of the five datasets tested.**

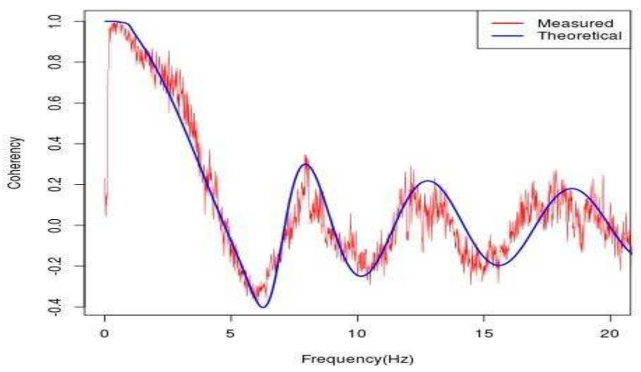


Fig. 1: Theoretical coherency curve.

I. Introduction

A model for characterising the soil shear-wave velocity profile and liquefaction potential of a site during an earthquake needs to be autonomously generated as human analysis is laborious and standard regression techniques ineffectual. The Christchurch area of New Zealand (NZ) suffered a series of earthquakes and aftershocks between 2010 and 2012 that highlighted the influence of local site conditions on patterns of earthquake damage; in particular, liquefaction of soils resulted in extensive ground, building, and infrastructure damage. Knowledge of the site shear-wave velocity is of particular importance for determining expected earthquake ground motions and the appropriate geological Site Class for earthquake-resistant design [1]. In addition, the shear wave velocity of the soil can be used to infer whether or not the site is prone to liquefaction. There is a known boundary at a shear wave velocity of 200 m/s where a material *will not* liquefy if it is above that point, however it *may* liquefy if below it [2][3].

Spatial Auto-Correlation (SPAC) is a non-invasive site investigation technique, for simple sites comprising horizontal soft sediments over a stiff layer, which uses a spatial array of seismic recorders to measure ambient surface wave vibrations at a particular site [3][4][5][6][2][7]. The arrays can be deployed quickly in an urban setting without the need for consent such as that required for drilling. The instruments record vibrations for approximately half an hour, and the subsequent data analysis and interpretation can be done in little more than a day. This means that the classification of a site can be determined relatively quickly compared to alternative methods [3][4].

Surface wave tests take advantage of the dispersive nature of *Rayleigh* waves, and knowledge that variation of phase velocities with frequency results from the variation of shear wave velocities with increasing depth [5][6]. Processing of these records can allow determination of shallow shear wave velocity profiles and the thickness of sub-surface layers. The higher the shear wave velocity, the ‘more solid’ the ground is, and the less prone to the ‘shaking up’ of porous spaces that causes liquefaction [3][7]. A dispersion curve is determined from a series of calculations based the wave field data recorded by the sensors on the ground surface (the *coherency* of the measurements from the recorders). This represents the way in which surface waves spread out as they travel across the surface of the earth, and a Bessel function is applied during the *inversion* process. This produces a theoretical coherency curve which is compared to the field-measured coherency to give a goodness-of-fit of an estimated soil layer model. Figure 1 shows a field-measured coherency plot, with the measured data shown in red and the theoretical coherency in blue.

The known underlying physical model means that regression techniques that simply produce arbitrary models will not be meaningful, as the evolved model must represent the soil layers in a realistic manner. Constraints must be enforced in various stages of the evolutionary process, for example model initialisation and evolutionary operations, which are difficult to capture in an error-based fitness function. The number of layers is not known *a priori* so must be determined as part of the model, and is currently part of an operator-driven visual evaluation process.

A. Motivations

Modelling of the surface layers is currently performed by feeding an estimated model of the surface layers into modelling software [8] with the layer structure being varied until an optimal match is obtained. This process is performed using the simplex algorithm [9] with intensive operator intervention being required.

Whereas the rich program structure of Genetic Programming (GP) has been used successfully to evolve solutions to Symbolic Regression (SR) tasks [10][11][12], it has not previously been applied to the Spatial Auto-Correlation (SPAC) domain. The layer model used fits particularly well into the structure of PLGP [13][14][15][16], a system that has not previously been applied to the SR domain.

B. Goals

This research will develop a working proof-of-concept for ESPAC, a system that will handle the model estimation process using PLGP to fit the data model within the known constraints. The goal is to shift the burden of analysis of SPAC data from a labour-intensive, operator-based process to one where an intelligent system can produce good quality candidate solutions for the operator to review. The specific objectives of the project are as follows.

- Define the problem as one of SR and extend PLGP to the regression domain.
- Use the existing Computer Programs in Seismology (CPS) modelling software for the calculation of Rayleigh wave dispersion.
- Provide curve matching results comparable to those achieved by a domain expert.

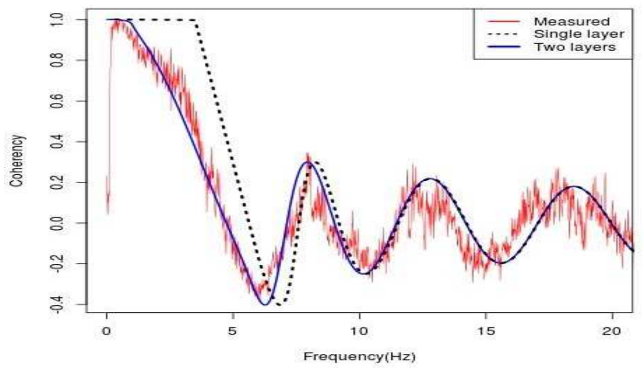
C. Organisation

The rest of the paper is organised as follows. Section II provides further background information on seismology and EC topics, and Section III describes the design of the solution presented by this project. Section IV gives details of the experimental setup, with the results described in Section V. Analysis is provided in Section VI and conclusions are made in Section VII.

II. Background

A. Spatial Auto-Correlation

Liquefaction is the process that leads to soil's sudden loss of strength, and is most commonly caused by the ground shaking during a large earthquake. If sands and silts, which are loose in the ground, sit below the water table, the space between the grains is filled with water. When an earthquake occurs, the shaking of the ground causes the grains to compress the water-filled space, forcing the water to build up pressure until the grains 'float'. At this point, the soil has liquefied and lost its strength, and behaves as a fluid. It cannot support the weight of whatever is lying above it and is forced into any gaps above it as surface materials 'sink'.



Model	Thickness (m)	Velocity (m/s)
Single layer	8.6	123.5
	–	–
Two layers	8.6	123.5
	136.5	287.7

Fig. 2: Refining the curve by adding a layer.

Spatial Auto-Correlation (SPAC) is a method of using array-based micro-tremor records to evaluate the *shear-wave velocity* of a soil column underlying a site [17][18][19][20]. Shear wave velocity of a granular material is related to the porosity of the material, such that a low shear wave velocity is associated with a great deal of pore space. A layer is likely to liquefy upon being shaken if it has a low shear wave velocity and the pore spaces are filled with water. [3][4][7].

The measurement of fundamental-mode *Rayleigh* waves which are present in micro-tremors allows the construction of a dispersion curve and determination of a shear wave velocity profile through the inversion process [3][4][7]. The *site period* can also be determined when SPAC is used in conjunction with Horizontal-to-vertical Spectral Ratio (HVSR) [21][22][20].

The Computer Programs in Seismology (CPS) software suite consists of a number of modular programs for the analysis of seismic data [8]. An estimation of the earth model is presented to the CPS suite, consisting of one or more layers represented by values for thickness and shear wave velocity of each layer. The CPS suite calculates a theoretical dispersion curve which is then passed to the inversion process, resulting in a theoretical coherency curve which is matched for error against the field measured data.

The heuristic approach currently used by GNS Science [3] successively introduces layers above or below an initial layer until adding layers no longer improves the model. Low frequencies are corrected by adding deeper layers, and high frequencies are corrected by adding a surface layer. This process continues using the simplex algorithm to call CPS with

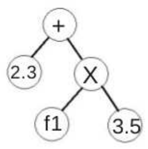


Fig. 3: GP Program.

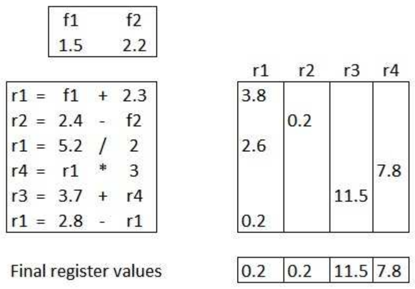
candidate layer models, iteratively refining the model until no further improvement can be made [3]. Figure 2 shows an example of the fitting process, with the theoretical coherency produced by a single layer shown by a black dotted line, and a two layer model shown by a solid blue line. Beneath the figure, the values for the layer models show that the two layer model has had a layer added below the first layer.

B. Genetic Programming for Symbolic Regression

Regression tasks are those in which the objective can be represented by a curve, and the objective is to estimate a model that best fits that curve. The goals of this project can be easily viewed as a regression task in which the measured coherency represents the target curve and the theoretical coherency is provided by existing seismology software. The goodness of fit of this theoretical curve represents the fitness function for an evolutionary process which will evolve layer models in a guided stochastic process.

Genetic Programming (GP) is an effective evolutionary approach to solving regression problems, due largely to the structure of the programs that enables complicated functions to be evolved from mathematical operators, random constants, and features from the data [10]. Each program is evaluated against some *fitness measure* to determine their effectiveness at solving the task and therefore their suitability as donors for the next generation. Mutation and crossover operations are applied to produce a child generation from stochastically selected parents, and elitism preserves the best solutions unmodified. A GP program iterates through each data point in the input set and calculates an output, representing the model that is to be matched to the objective curve. The *fitness* of that program is typically calculated by Root Mean Squared Error (RMSE) or some other goodness-of-fit measure. After a predefined number of generations, the best program that has been evolved is chosen to represent the best model discovered by the system, and that program represents the function that can be applied to the input to give the best estimation for the output. The most common form of GP is Tree Based GP (TGP) [10], where programs are structured similarly to Lisp S-expressions. An example of a simple TGP program is shown in Figure 3.

Linear Genetic Programming (LGP) consists of a linear sequence of instructions using a set of registers that may be re-used during execution of the program and give multiple outputs upon completion [12][23][24][25]. Similar to assembly language, each instruction may use up to two registers as input and apply the same genetic operators as TGP to produce a single register as output. The strengths of LGP are that values that have been computed by one instruction can be used by



f1	f2
1.5	2.2

	r1	r2	r3	r4
r1 = f1 + 2.3	3.8			
r2 = 2.4 - f2		0.2		
r1 = 5.2 / 2	2.6			
r4 = r1 * 3				7.8
r3 = 3.7 + r4			11.5	
r1 = 2.8 - r1	0.2			
Final register values	0.2	0.2	11.5	7.8

Fig. 4: LGP Program.

another instruction later in the program execution, and that LGP can easily produce multiple outputs, each representing a particular class. This makes LGP a powerful technique that has been shown to outperform TGP in multi-class classification [24][25]. However, every time a result is reused an *instruction dependency* is introduced, which means that a relatively minor modification to one instruction (in evolution) may then cascade throughout the program by altering the value in a register that is used by multiple subsequent instructions. As large and random changes are known to disrupt the performance of GP, the strength of LGP is also a potential shortcoming.

An example of an LGP program is shown in Figure 4, where a simple program performs a classification task on a data instance comprised of two features *f1* and *f2*. The program is executed from top to bottom, and at each point where a register is assigned to, that value is displayed in the appropriate column on the right. The final register values for $r1 \cdots r4$ are listed, and in a classification task the index of the highest register value corresponds to the classification made for that data instance (in this example, class 3). There are several instruction dependencies, where a previously computed value is re-used from one of the registers.

Parallel Linear Genetic Programming (PLGP) was developed to limit instruction dependencies between instructions by controlling the interaction between them [13][14]. Here, n instruction sequences, called *factors*, are evaluated independently to give *n* register vectors which are then summed to give a single result vector. Each program factor has its own vector of registers which are initialised to zero before execution and are not passed to the following factor. This prevents instruction dependencies between factors, and limits the potential number of instruction dependencies to the number of instructions in each factor. By regarding each program as an *ordered list* of factors, crossover is limited to *equivalent factors*, i.e. those that occupy the same list position in both programs. This constraint creates Enforced Sub-Populations (ESP) where genetic material may flow within subpopulations but not between them [26]. The concept of ESP was further extended in the Co-operative Coevolution Parallel Linear Genetic Programming (CC-PLGP), Blueprint Search Parallel Linear Genetic Programming (BS-PLGP) and Hill Climbing Parallel Linear Genetic Programming (HC-PLGP) architectures, where subpopulations of partial solutions were evolved in isolation

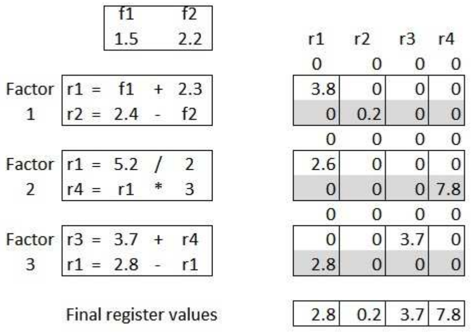


Fig. 5: PLGP Program.

from each other, and a complete solution was formed by combining a number of partial solutions [13][15][16].

An example of a PLGP program is shown in Figure 5, and closely resembles the LGP program shown in Figure 4. The program has been broken into three factors, and before each factor is executed the register values have been reset to zero. The final register values are found by adding the vectors that result after each factor is executed (which are shaded). Breaking the program into factors has reduced the number of instruction dependencies, and in this case has changed the classification from class 3 to class 4. PLGP has been shown to be an effective method for classification [13][14][15][16], though it has not previously been applied to regression tasks.

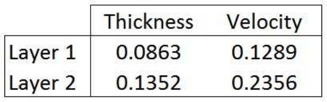
III. Evolutionary Spatial Auto-Correlation (ESPAC)

ESPAC has been designed to perform a very specific real-world task, and accordingly has a number of unusual modifications from what could be considered the norm. For this domain, there is a paradigm shift when considering how best to represent the population of PLGP programs. To help define this, careful consideration must first be given to exactly how ESPAC will fit into the existing process of model creation.

The required functionality of the CPS *modal summation* module [8] has been retained, and is implemented in this project as a Java Native Interface (JNI) library that is called from the Java ESPAC system. This modified library will hereafter be referred to as *libCPS*.

ESPAC evolves *models* which represent some estimated layer profile and will be checked for goodness of fit by calculating and inverting the dispersion curve. A population of randomly generated models is initially created by the system, and these are evaluated and evolved by the system in place of GP programs. Each model is to be evaluated by using libCPS to calculate the theoretical coherency for that model, with the field measured coherency as the target value. An example of an ESPAC model is shown in Figure 6, where the model is similar in structure to a PLGP program with layers in place of factors.

LibCPS represents the 'execution' phase of the evolution, where candidate models produce an output which is measured for goodness of fit against the target curve. High fitness models are identified as useful donors for the next generation and



	Thickness	Velocity
Layer 1	0.0863	0.1289
Layer 2	0.1352	0.2356

Fig. 6: ESPAC Layer model.

evolution is performed, where the evolution operators focus on refining existing good solutions by *adjusting* the values of the layer model. The standard GP operations of mutation and crossover are complemented with addition (ADD), subtraction (SUB), division (DIV), and multiplication (MUL).

A. Redefined Structure of Candidate Solutions

PLGP programs consist of a number of factors, each containing a number of assignments to registers and the result vector from each factor is summed (as shown in Figure 5). This structure is not useful for this domain, as the calculations are performed by libCPS and not the candidate solutions. ESPAC instead uses *layer models* that represent an estimate of the soil profile. Each model consists of a number of layers which are represented by a *thickness* and *shear wave velocity*. The number of layers is not known *a priori*, the determination of which is part of the search process. A simplification has been made for this project in which all models are defined as having two layers, allowing the remaining parts of the system to be developed to the working proof of concept stage.

The Enforced Sub-Populations (ESP) that is part of the PLGP architecture is of particular interest, as is the specialisation of partial solutions that results. By basing the layer model on the PLGP architecture, crossover is constrained so that genetic material can only be exchanged between layers that occupy the same position in their respective models. For example, the top layer of one model will only combine with the top layer of any other model. The effect of this ESP will be to refine each layer in isolation from the others. Initial experimentation suggested that the structure of the models, having only two layers and four values, was prone to disruption if the subpopulations were not enforced, and when the system is extended to allow a more dynamic number of layers in the model, the constraints will allow the system to search for partial solutions more effectively [27][28][29][30].

B. Redefined Genetic Operators

GP evolution tends to consist of crossover, mutation, and elitism operations to evolve useful solutions, and a typical system may use (for example) 70% crossover, 20% mutation, and 10% elitism during the evolutionary process. Due to the small number of values that make up the layer models, i.e. two values for each of two layers, some consideration is necessary to prevent layer models from being unduly disrupted by a process that encouraged replacement rather than refinement. Initial experimentation showed that an evolutionary process that was predominantly driven by crossover tended to disrupt diversity, as the fittest individuals had their values propagated throughout the population quickly. Due to the underlying geophysics of the SPAC process, replacing a value in a model by crossover may make the model nonsensical and invalidate the values that may have been useful.

A new range of 'modification' operators was used by ESPAC to encourage layers to be 'fine-tuned', and they operate alongside the traditional mutation, crossover, and elitism. These new operators consisted of addition (ADD), subtraction (SUB), multiplication (MUL) and division (DIV), and they were bounded by range parameters. As ADD and SUB are similar operations they shared the same range as each other, and MUL and DIV shared the same range as each other. For example, with an ADD/SUB range of 0.3, the ADD operator would add some random amount in the range [0.0, 0.3], and SUB would subtract an amount from the same range. With a DIV/MUL range of 0.3, the MUL operator would multiply the value by a random number in the range [1.0, 1.3] and DIV would divide by a number in the same range. Note that protected division is not required as there can be no division by zero. The modification operators were applied to only a single value, e.g. the thickness of the top layer, or the velocity of the bottom layer. This was intended to allow the system to react to small modifications, evaluating the fitness of individual changes during the selection process.

Mutation replaced a single value with a new, randomly generated value, e.g. replacing the thickness of the top layer with a new value. As with initial model generation, the range of these random values was defined by a parameter at the beginning of the experiment. Crossover was also applied to a single value, and this was constrained by ESP to only permit material to be exchanged between equivalent layers. For example, model A and model B may exchange the values for thickness of the top layer, but the thickness may not be from the top layer of model A and the bottom layer of model B. Elitism simply preserved the best solutions unchanged in the child generation.

C. *Evolutionary Process*

The evolutionary process for this project has not been greatly modified from the norm. A population of models is evaluated and the fittest individuals are selected as donors using tournament selection. The resultant models are added to the elite models in the child population until it reaches its specified size. Once the evolutionary process is complete and the child population is full, the models are again evaluated and the evolutionary process continues.

D. *Redefined Execution / Fitness Environment*

A traditional GP approach to regression uses the features from the dataset as input and executes each of the programs in the population, using that program's output to determine the fitness. The execution phase in ESPAC is instead performed by libCPS, using the layer models as input and the resultant theoretical coherency as the curve we are attempting to fit to the measured data. In performing the error matching, a number of fitness functions were considered, and due to the increased importance of the lower frequencies of the coherency curve [3] the fitness function was weighted accordingly. Initial experimentation showed better performance was achieved by the following fitness function than RMSE. The error for each data point is the absolute value of the error for that point, divided by the frequency modified by a weighting parameter α. The square root of the mean of those errors is ESPAC's fitness function.

$$\sqrt{\sum_{i=1}^{n} \frac{\mid \hat{\theta}_i - \theta_i \mid}{n \cdot \alpha \cdot frequency_i}}$$

$\hat{\theta}$: Theoretical coherency
θ : Target coherency
$frequency$: The frequency for the given data point
α : Weighting parameter | set at 0.02
n : Number of data points

IV. Experimental setup

A. *Datasets*

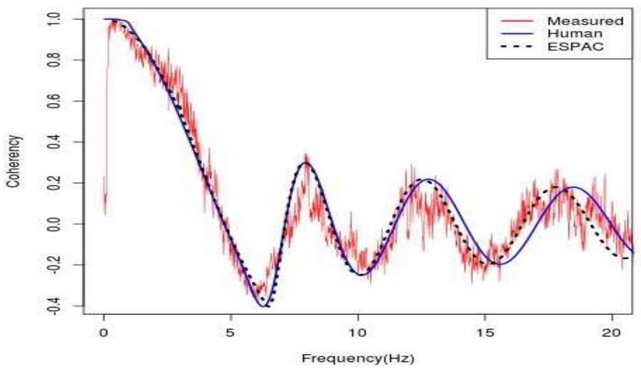
Datasets are defined by a set of field measured coherency values and the aperture size, or distance between the sensors that recorded the data at that site. There are five sites used for this project, simply referred to as sites 1 – 5, and all are from the Christchurch area of NZ. Sites 1 and 2 were determined by a human expert to be best represented by two layers, and the best human defined model for sites 3 – 5 contained only one layer.

1) Evolutionary Parameters: A population size of 1000 was used, and due to time constraints, evolution was limited to 50 generations. Tournament selection was used with a tournament size of 5. Crossover was set to 30%, mutation 10%, and elitism 10%. ADD and SUB were set to 15% each, and DIV and MUL were both set to 10%. This balance of operators was selected to encourage the adjustment of layer values rather than the usual crossover-driven evolutionary process. A small amount of mutation was allowed to prevent the evolution becoming trapped in local optima, and elitism was anticipated to provide a degree of backtracking by ensuring many of the best solutions remained in the population.

2) ESPAC Parameters: The random layer generation range was 0.01 for thickness (up to 10m), and 0.5 for velocity (up to 500m/s). These parameters were set after observation of a number of models that had been created by a human expert. The random range for ADD and SUB were set to 0.1 (100m) for thickness and 0.5 (500m/s) for velocity, to provide values for modification that would encourage gentle adjustment. The range for DIV and MUL was set to 0.5 for both thickness and velocity, resulting in a division or multiplication in the range [1.0, 1.5].

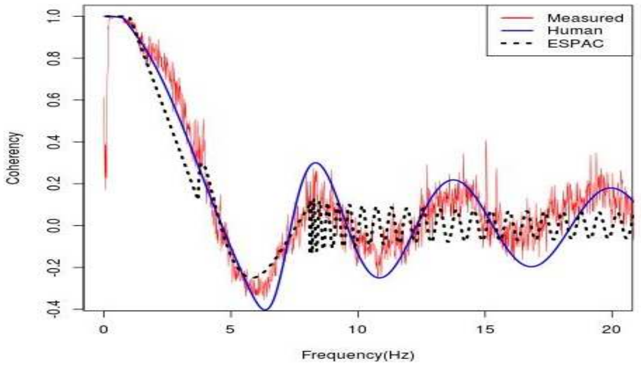
V. Results

In this paper, the objective was to adapt PLGP to solve a real-world problem and test the ability of ESPAC to produce a comparable model to that of a domain expert. Rigorous statistical significance testing over 30 or more runs was not



Model	Thickness (m)	Velocity (m/s)	Fitness
Human-simplex	8.6 136.5	123.5 287.7	1.6666
ESPAC	861.8 15.7	275.3 111.0	1.6136

Fig. 7: Human and ESPAC models for site 1. Low fitness is preferred.



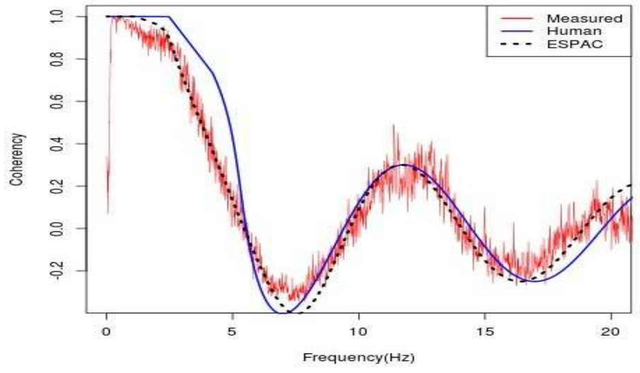
Model	Thickness (m)	Velocity (m/s)	Fitness
Human-simplex	9.0 163.0	133.5 273.1	1.4217
ESPAC	1.3 1.8	279.8 15.0	1.6136

Fig. 8: Human and ESPAC models for site 2. Low fitness is preferred.

required as the goal was not to measure the ability of the method to solve regression tasks in general. Due to time and resource constraints, there were only five replications of the experiment performed on each site and the overall most fit model has been represented in the results. An additional model has been presented for site 4 to provide interesting context. It is important to note that although the ESPAC curves are displayed using a dotted black line, the dots do not represent individual data points.

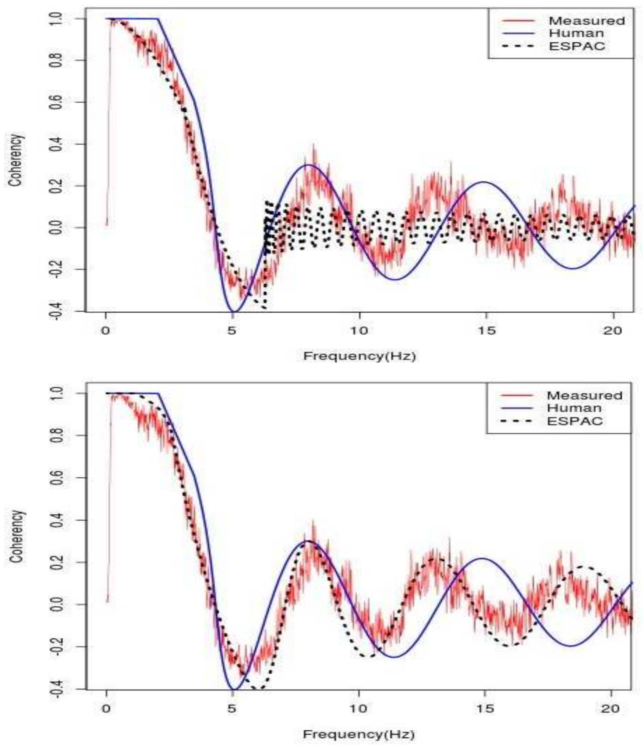
Site 1: The ESPAC model fits the field measured coherency very closely throughout the curve in Figure 7, and was evolved in 49 generations. The table below the plot shows that the two models were very different from each other, with ESPAC defining a top layer that was ten times thicker than the human-simplex model. The fitness given in the rightmost column confirms a good representation of the curves.

Site 2: As shown in Figure 8, this is an example where ESPAC has performed badly. There is rapid oscillation of the curve beyond 8 Hz, and the curve also matches poorly below that range. The models shown in the accompanying table show that the fitness value is representative of the poor fit. The thickness of the layers for the ESPAC model, which was evolved in the 17th generation, show that ESPAC has simply not been effective at solving this problem.



Model	Thickness (m)	Velocity (m/s)	Fitness
Human-simplex	16.1 –	164.6 –	1.5573
ESPAC	9.8 41.9	157.5 245.4	1.5212

Fig. 9: Human and ESPAC models for site 3. Low fitness is preferred.



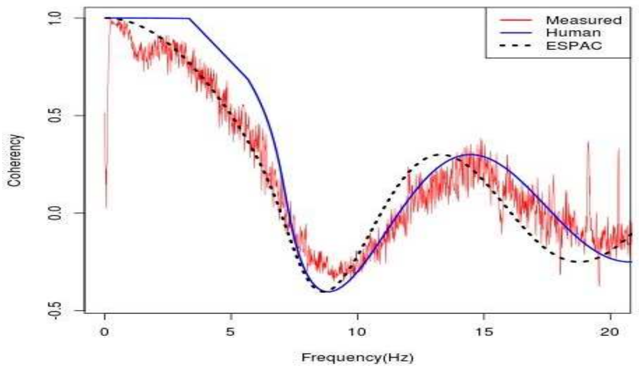
Model	Thickness (m)	Velocity (m/s)	Fitness
Human-simplex	17.4 –	146.9 –	1.7845
ESPAC Model 1	736.4 2.0	284.4 13.0	1.6786
ESPAC Model 2	8.6 41.0	126.5 245.9	1.7481

Fig. 10: Site 4. Human and ESPAC model 1 (top) and model 2 (bottom).Low fitness is preferred.

Site 3: Figure 9 shows that ESPAC has fitted the coherency curve tighter than the human-simplex model, and this is reflected in the fitness value shown in the accompanying table. This was the first example of a site where the human-simplex method restricted the model to a single layer, and is compared to a two layer ESPAC model that was evolved in 18 generations. The table shows that the shear wave velocities for the top layer of both models were very similar.

Site 4: In Figure 10 two plots are shown to illustrate the complexity of the domain, and is another example of a single layer human-simplex model. ESPAC model 1, shown in the top plot, had the overall best fitness value, evolved after 30 generations. Model 2, shown in the bottom plot, had a lower fitness, and was evolved after 24 generations. Model 1 has an very thick top layer and thin bottom layer. The plots clearly show that, although the fitness of model 1 is the lowest, the oscillations make it a poor fit. Model 2, however, fits well and its fitness relative to the human-simplex model is reasonable.

Site 5: Figure 11 is another instance of a single layer human-simplex model. The ESPAC model fits the lower frequencies below 8Hz very well, and the human-simplex model fits better over 10Hz. The ESPAC model has a better fitness value, reflected by the importance given to lower frequencies. The top layer thickness and velocity are similar for both models, as shown in the accompanying table. This ESPAC model was evolved in 41 generations.



Model	Thickness (m)	Velocity (m/s)	Fitness
Human-simplex	19.3 –	268.7 –	1.6666
ESPAC	14.3 1309.6	264.0 471.4	1.6136

Fig. 11: Human and ESPAC models for site 5. Low fitness is preferred.

VI. Analysis

The results from sites 2 and 4 demonstrate that it is possible to evolve models that clearly do not fit the data. The rapid oscillations that are a feature of the evolved models is representative of many models that are randomly generated during the initialisation process. Whereas it would generally follow that such oscillation could be caused by overfitting, it is perhaps more indicative of the difficulty in finding solutions that fit the constraints of the real world geophysical model for this domain. It is immediately obvious to the observer that these models fit poorly, but this judgement has not been effectively captured in the fitness function, as indicated by the good fitness values given to model 1 of site 4.

There are two approaches to the resolution of this problem, and they can be broadly defined as matters of either *constraint* or *evaluation*. Using constraints to address the issue would involve preventing nonsensical solutions from being evolved. In part, this is the approach taken when using operator experience, being inefficient to expend effort on invalid models. Some restraint must be exercised when enforcing constraints on the evolutionary process, as the strength of GP is the ability to discover solutions that a human may not conceive. The net effect of this is the potential for the system to discover new relationships and rules rather than being limited to confirming those which are known. Improving the evaluation would primarily involve a more effective fitness function, which could include the penalisation or promotion of models according to their structure. Complementing the fitness function in this way differs from constraints in that nonsensical models are allowed to evolve, but their fitness is penalised. This retains diversity in the population, as well as retaining the ability of GP to ‘stumble upon’ good solutions.

The results for site 1 (Figure 7) show two models that are very similar in fitness and curve, although are very different in composition. If the HVSR matching process were applied to these models, it would be more clear which of these models was more appropriate. A feature of the poorly fitting ESPAC models for sites 2 and 4 is a very thin second layer, which may have been exposed earlier using the HVSR technique.

The experimental design featured a large population of models and a small number of evolutionary generations. Due to the fact that only a small number of models are high fitness parents, a better use of computational resources may have been to reduce the population size and evolve them over a larger number of generations. This would have resulted in a larger number of genetic operations over a smaller number of models rather than a large population that may have been largely stagnant.

VII. Conclusions

The ESPAC system has been effective at performing a real-world regression task, which is a major contribution for PLGP. However, the problem has not been defined as well as it might as the models may be nonsensical or unrealistic in the real world, and evaluation of them requires more considerations than simple error matching. Analysis of these processes and results by a domain expert is essential to re-evaluate some of the assumptions and simplifications that have been made, as well as investigating the effect of the parameters and constraints that have been introduced. The conclusions with regard to the specific objectives are as follows.

- PLGP has been successfully extended to perform the regression task. The definition of the task will benefit from further analysis of the results by a domain expert.
- *libCPS* has successfully captured the functionality of CPS and enabled ESPAC to use the existing modelling software. The execution phase of the process will benefit

from increasing the efficiency of libCPS, perhaps by parallel processing.

- The models for sites 1, 3, and 4 (model 2) match the objective curve more closely than the human-simplex method, and site 5 is comparable. ESPAC was unable to effectively evolve a model for site 2.

The number of layers was set to two for this project, although this value is to be discovered by the search mechanism in future work. By allowing a dynamic number of layers we may allow a much better estimation of layer models and better capture domain knowledge. A suggested approach would be to evolve a population of single layer models for the first *n* generations, and then spend another *m* generations attempting to find suitable layers to refine the models. A single layer model's fitness could be cached, and another layer is added only if it improves fitness, allowing for backtracking.

The co-use of SPAC and HVSR would greatly aid the process by refining the thickness of a second layer. This effectively reduces the range of one of the variables and allows the search to concentrate on the other unknowns.

As ESPAC is the first Evolutionary Computation (EC) system applied to this problem, it should be compared to other EC approaches, i.e. Genetic Algorithms, TGP, LGP, or Neural Networks. This comparison would evaluate the ability of EC in general to produce effective solutions to this domain in place of human judgement.

To the best of our knowledge, this is the first application of GP to this problem domain. While many aspects are worth further investigation, initial experiments show that the results are comparable to human experts and there is great potential to apply GP to difficult problems in this domain.

Acknowledgements

This work is supported in part by the research grants E1983/2541 from Earthquake Commission/GNS Science and the University Research Grant 200457/3230 at Victoria University of Wellington.

References

[1] Standards New Zealand 2004, "Structural design actions part 5 earthquake actions new zealand," New Zealand Standard NZS 1170.5:2004, 2004.

[2] R. D. Andrus and K. H. I. Stockoe, "Liquefaction resistance of soils from shear-wave velocity," *Geotechnical and Geoenvironmental Engineering*, vol. 11, pp. 1015–1025, 2000.

[3] W. Stephenson, P. Barker, Z. Bruce, and R. Beetham, "Immediate report on the use of micotremors (spac measurements) for assessing liquefaction potential in the christchurch area," GNS Science, Tech. Rep., 2011.

[4] R. Beetham, W. Stephenson, P. Barker, and N. Perrin, "A non-invasive site investigation method for determining site class from microtremor records," in *Proceedings of the 11th IAEG Congress*, 2010.

[5] S. Nazarian and K. H. Stokoe II, "In situ shear wave velocities from spectral analysis of surface waves," in *Proceedings of 8th Conference on Earthquake Engineering*, vol. 3, 1984, pp. 38–45.

[6] K. Stokoe II, G. Wright, A. James, and M. Jose, "Characterization of geotechnical sites by sasw method," in *Geophysical characterization of sites*, 1984, pp. 15–25.

[7] W. Stephenson, P. Barker, and R. Beetham, "Microtremor based determination of shear-wave velocity of soft soils using the spac method," GNS Science, Tech. Rep., 2011.

[8] R. B. Herrmann, "Computer programs in seismology," Saint Louis University, USA, 2002.

[9] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[10] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, Mass.: MIT Press, 1992.

[11] J. Niehaus and W. Banzhaf, "Adaption of operator probabilities in genetic programming," in *Genetic Programming, Proceedings of EuroGP2001, volume 2038 of LNCS*. Springer-Verlag. 2000, 2001, pp. 325–336.

[12] M. Brameier and W. Banzhaf, "Evolving teams of predictors with linear genetic programming," in *Genetic Programming and Evolvable Machines*. Kluwer, 2001, pp. 381–407.

[13] C. Downey and M. Zhang, "Parallel linear genetic programming," in *Proceedings of the 14th European Conference on Genetic Programming*, 2011, pp. 178–189.

[14] C. Downey, M. Zhang, and J. Liu, "Parallel linear genetic programming for multi-class classification," *Genetic Programming and Evolvable Machines*, vol. 13, no. 3, pp. 275–304, 2012.

[15] A. Scoble, M. Johnston, and M. Zhang, "Local search in parallel linear genetic programming for multiclass classification," in *Australasian Conference on Artificial Intelligence*, 2012, pp. 373–384.

[16] A. Scoble, "Hill climbing search in co-evolutionary parallel linear genetic programming for multi-class classification," Honours report, Victoria University of Wellington, NZ, 2012.

[17] K. Aki, "Space and time spectra of stationary stochastic waves, with special reference to microtremors," *Bull. Earthquake Res. Inst. Univ. Tokyo*, vol. 35, pp. 415–456, 1957.

[18] ——, "A note on the use of microseisms in determining the shallow structure of the earths crust," *Geophysics*, vol. 30, pp. 665 – 666, 1965.

[19] H. Cox, H. Lai, and K. Bell, "Spatial correlation for directional sensors in arbitrary noise fields," in *Proceedings of the 43rd Asilomar conference on Signals, systems and computers*, ser. Asilomar'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 459–463. [Online]. Available: http://dl.acm.org/citation.cfm?id=1843565.1843668

[20] P. Apostolidis, D. Raptakis, and K. Pitilakis, "The use of microtremors for the definition of soil properties and bedrock depth in an urban area," 2004.

[21] Y. Nakamura, "A Method for Dynamic Characteristics Estimation of Subsurface using Microtremor on the Ground Surface," *Quarterly Report of Railway Technical Research Institute (RTRI)*, vol. 30, no. 1, 1989.

[22] J. Lermo and F. J. Chavez-Garcia, "Are microtremors useful in the site response evaluation?" *Bulletin Seismological Society of America*, vol. 84, pp. 1350 – 1364, 1994.

[23] M. Brameier and W. Banzhaf, "A comparison of linear genetic programming and neural networks in medical data mining," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 1, pp. 17–26, 2001.

[24] C. Fogelberg and M. Zhang, "Linear genetic programming for multi-class object classification," in *AI 2005: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, S. Zhang and R. Jarvis, Eds., 2005, vol. 3809, pp. 369–379.

[25] G. Olague, E. Romero, L. Trujillo, and B. Bhanu, "Multiclass object recognition based on texture linear genetic programming," *Applications of Evolutionary Computing*, pp. 291–300, 2007.

[26] F. Gomez and R. Miikkulainen, "2-d pole balancing with recurrent evolutionary networks," in *Proceedings of the International Conference on Artificial Neural Networks*, 1998, pp. 425–430.

[27] M. Potter and K. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science, Y. Davidor, H.-P. Schwefel, and R. Milnner, Eds., vol. 866, 1994, pp. 249–257.

[28] ——, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation*, vol. 8, no. 1, pp. 1–29, 2000.

[29] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[30] D. E. Moriarty and R. Miikkulainen, "Forming neural networks through efficient and adaptive coevolution," *Evolutionary Computation*, vol. 5, pp. 373–399, 1997.