

TESTING DISCIPULUS LINEAR GENETIC PROGRAMMING SOFTWARE ON REAL-WORLD ENVIRONMENTAL ENGINEERING CHALLENGES

(Draft Working Copy – Pre-Released for Collaborative Information Sharing Purposes Only)

Larry M. Deschaine, PE
Science Applications International Corp.
360 Bay Street Suite 200
Augusta, GA 30909
+1-706-724-5589
Larry.M.Deschaine@alum.mit.edu

INTRODUCTION

Genetic Programming (GP) is a machine learning technique that writes computer programs, automatically. Although individual researchers used GP techniques in the 1960's and 1970's, GP emerged as a distinct discipline in 1992.⁶³ Since that time, over one thousand academic studies have been published in the field and, in 1998, commercial GP software – Discipulus – reached the market.²⁰ Discipulus is an extremely fast, linear GP system written for the Wintel platform. It runs about two orders of magnitude faster than first-generation GP software. Discipulus generates C/C++ and assembly language programs that map process inputs to outputs.

This paper reports results from testing Discipulus on three very difficult environmental engineering problems. It also reports the results of attempts to “fool” Discipulus into inducing a false relationship by memorizing training data. The test problems presented here all involve real-world environmental engineering data from projects that have million-dollar-plus implications. The problem domains include waste incineration facility simulation, soil stabilization mix design, and an *in-situ* technique for inferring geotechnical properties.

In brief summary, the Discipulus GP software generated high-precision models for each of these problems, with good generalization properties. The evolved solutions performed at least as well as – most often significantly better than – solutions derived from well-conducted neural network studies or other analysis techniques applied to the same problems. The evolved models have been or are being used to develop process prediction or control algorithms.

All analyses below have been conducted by, or under the direction of, a licensed Professional Engineer.

MACHINE LEARNING

Machine Learning is a process that maps a set of input data to known target output data. For example, if the true mapping function is described as $y=f(x_i)$, the approximated mapping function (derived by the computer) is described as $y=\hat{f}(x_i)$, where x_i is the i^{th} input variable. The evolved program is, therefore, a knowledge representation of the underlying data information, which can be used to predict and optimize processes, as merited. The goals of deploying the machine learning technique on engineering challenges are:

- Determine which inputs (x_i) contain the most relevant information to describe the output information. This is known as data mining, and
- Develop the mapping function $y=\hat{f}(x_i)$ that relates the input information to the output information. This is known as knowledge discovery.

Using machine learning to map a series of inputs to outputs where the relationships are unknown provides a significant benefit in many areas of science and industry, including dynamic and batch manufacturing process control optimization and remote exploration, as discussed below, just to name a few.

THE SPEED PROBLEM IN MACHINE LEARNING

Almost all Machine Learning technologies are computationally intensive, whether implemented as genetic algorithms, genetic programming, simulated annealing, analytical neural networks, or decision trees. Often, the speed of obtaining a solution, or more precisely the lack of speed, is a major hindrance in machine learning deployment. Most machine learning systems are forced to limit their search for good solutions because of speed limitations. More important, few machine learning systems can afford to perform hundreds of runs to produce a single “great” solution to a problem. Yet, hundreds of runs is often precisely what is needed to generate a high-precision solutions to difficult engineering problems.⁷

The Discipulus GP system uses AIMLearning™ Technology. “AIM” stands for “Automatic Induction of Machine Code”. AIMLearning and Discipulus deal with the machine learning speed problem by brute force – they evaluate solutions in processor specific machine code.¹⁻¹⁴ Machine code implementations are sixty to two-hundred times faster than other methods of genetic programming.^{1-14, 64} To get a sense of how much faster this technology is than other machine learning systems, running an AIMLearning based system on a *single* computer is equivalent to running other machine learning software on about *one hundred* computers, operating in parallel. Viewed another way, for very complex problems, AIMLearning may take a few days on a single computer to generate a reasonable solution, other techniques would be require months or years of run-time on the same computer.

While a speed-up of sixty to two-hundred times would seem, at first blush, to be overstated, it has been repeatedly duplicated¹⁻¹⁴ and has been independently verified by researchers at Jet Propulsion Laboratories.⁶⁴ The author’s personal experience with Discipulus is consistent with a speedup of this magnitude. Indeed, the speed of AIMLearning based systems like Discipulus was one of the principal reasons for choosing Discipulus for this study.

From an engineering perspective, a few days is acceptable, a few months is not. In practice, we have found that good solutions to simple systems are evolved on the order of minutes to hours once the data sets are prepared and validated. For complex systems, like the incinerator data discussed in Solution Set-#1, it took Discipulus about twenty-four hours to solve the carbon dioxide equation on a dual processor machine. This speed allows the analyst to ability to make many more runs to investigate relationships between data and output, assess information content of data streams, uncover bad data or outliers, assess time lag relationships between inputs and outputs, and the like. The analyst can also evaluate more feasible solutions than before possible. This results in better performing — more robust and optimal — final solutions to the challenge.

THE SELECTION OF GENETIC PROGRAMMING FOR THIS STUDY

Many researchers use Analytical Neural Networks (ANN) for machine learning applications.¹⁵⁻¹⁸ AIMLearning techniques could have been used to implement machine-learning techniques other than GP, such as ANN. Nevertheless, we selected GP as the technology of choice for this work for the following primary reasons:^{1-14,19}

1. **Transparency of Solutions.** Where a Professional Engineer must certify a project, GP solutions are usually more acceptable than ANN based solutions. GP solutions are computer programs that can be inspected, documented, evaluated, and tested. The investigator may examine the GP solutions to understand the nature of the derived relationship between input and output data.

This process often helps uncover relationships that were heretofore unknown. It also permits the engineer to understand *how* the GP-evolved solution works. This avoids solutions that are a good-fit, but physically non-sensible. By way of contrast, the ANN solution is essentially a black box when it comes to this level of critical analysis.

2. **Simultaneous Induction of Structure and Contents.** Genetic Programming evolves both the structure and the constants to the solution simultaneously. A comparable technique in the ANN approach is to use a Genetic Algorithm (GA) to optimize the structure of the neural network, then use

the back propagation technique to optimize its weights. This stepwise approach is more cumbersome and less efficient than GP's approach as well as being *extremely* slow.

3. **Input Variable Filtering.** Discipulus GP strongly discriminates between relevant input data and inputs that have no bearing on a solution.²⁰ In other words, Discipulus performs input variable selection as a by-product of its learning algorithm. By way of contrast, ANN systems have little capability in this regard – the work of variable selection is usually a manual job for the investigator.

Filtering out irrelevant inputs can be very useful – the investigator can examine the evolved programs and see which input variables have been included and to what extent. The investigator can also see which inputs have been omitted in the final solution. With this information, the solution can be compared to process knowledge and judged as to whether it makes sense or is a non-sensible mathematical abstraction. This is a very key attribute of GP when applied to mission critical applications that model physical processes.

4. **Generalization vs. Memorization.** The proprietors of Discipulus GP claim that it has been carefully designed to avoid overfitting and memorization, a problem that constantly plagues ANN developers. We were very curious to find out whether that claim would be borne out in actual runs on real-world engineering data.

THE GENETIC PROGRAMMING ALGORITHM

Genetic Programming is a form of machine learning that automatically writes computer programs. It uses the principle of Darwinian Natural Selection to select and reproduce “fitter” programs. GP applies that principle to a population of computer programs and evolves a program that predicts the target output from a data file of inputs and outputs.¹⁴ The programs evolved by GP—in this case C/C++ and assembly code—represents a mapping of input to output data.

Discipulus implements a steady state GP algorithm as follows:^{20,21}

1. Initialize a Population of Programs. Create a population of randomly generated programs.
2. Tournament Contest. Randomly select four programs from the population. Evaluate them for how well they map input data to output data. This step is known as the program “fitness” evaluation. Two programs are selected as winners, and the other two are tagged as losers.
3. Transform the “Winner” Programs. The two “winner” programs are then copied and transformed probabilistically by:
 - Exchanging parts of the “winner” programs with each other to create two new programs (crossover); and/or
 - Randomly changing each of the tournament winners to create two new programs (mutation).
4. Replace the “Loser” Programs. Replace the “loser” programs in the population with the transformed “winner” programs. The winners of the tournament remain in the population unchanged.
5. Iterate Until Convergence. Repeat steps two through four until a program is developed that predicts the behavior sufficiently.

THE FOUR PROBLEMS STUDIED IN THIS PAPER

The Discipulus GP system was tested with four data sets, including: (1) three real-world data sets representing three very difficult, albeit typical, environmental engineering problems; and (2) a chaotic time-series. The three environmental engineering data sets are:

- Dynamic Process Simulation: Manufacturing processes where the physics and chemistry is complex and not well understood (Solution Set # 1);
- Batch Manufacturing: Predicting the strength, permeability and chemical stability of process designed materials (Solution Set #2);
- Remote Exploration: Inferring material properties via remote sensing (Solution Set # 3).

In addition, we attempted to deceive the Discipulus into learning false relationships between inputs and outputs with a fourth solution set (the chaotic time-series data), which we refer to as the False Positive Relationship Protection data set.

SOLUTION SET #1 -WASTE INCINERATION FACILITY (Dynamic Process Simulation)

Predicting the output of an industrial process is critical for efficient optimal operation as well as compliance with environmental discharge permits. This process is so critical that approximately 10% of new plant construction capital outlays are spent on process control equipment.²²⁻²³

To test the Discipulus GP software, the performance of a waste incineration plant was modeled using real-time typical operational data collected hourly over a one-week period at the Consolidated Incinerator Facility (CIF) at the Department of Energy Savannah River Site (DOE-SRS). The CIF processes a variety of solid and aqueous waste, using a combination of a rotary kiln, a secondary combustion chamber, and an off gas scrubber system. The process was chosen for two reasons:

- (1) It is a very complex system consisting of variable fuel and waste inputs, high temperatures of combustion, and high velocity off gas emissions; and
- (2) We had previously attempted to model this process in a well-designed study with a good quality, commercial, neural network system. In that study, we failed to achieve commercially acceptable results.²⁵

Data for eleven input variables and twelve output variables, describing the CIF process, was collected as follows:

Process Inputs

- Rotary Kiln (lb./hr): Fuel Oil (flow), Liquid Waste (flow), Solids (air flow), Solids (flow), Solids (average flow), Fuel Oil (air flow), Liquid Waste (air flow), Aqueous Waste (flow)
- Secondary Combustion Chamber (lb./hr): Fuel Oil (flow), Fuel Oil (air flow), Steam (flow)

Measured Process Outputs

- Rotary Kiln (lb./hr): Temperature
- Secondary Combustion Chamber: Temperature, O₂ (percentage), CO₂ (concentration as PPM)
- Off-gas: 4 measurements of CO₂ (PPM), 2 measurements of O₂ (percentage) and 2 measurements of duct flow (scfm).

The results for temperature simulation of the Rotary Kiln Incineration process unit are shown on Figure 1, and the off-gas discharge rates (scfm) are presented in Figure 2. The Discipulus GP technique was able to simulate these output variables to within an average of 1.3% of their measured value, with no value exceeding a 9% deviation. Figure 3 shows the results for the carbon dioxide (ppm-CO₂) emissions for the last three (72 hours) burn cycles. Note that the GP solution was able to mimic this very complex and dynamically changing output variable quite well; R² is 0.993 and the total mass balance error was about 1%.

These results allow the prediction of air emissions from the process to be both computed and/or predicted from the input information alone. The importance of these results cannot be overemphasized. The potential impact for cost savings in Clean Air Compliance activities is very substantial.

These results can be combined with Enterprise Resource Planning information to optimize plant production operations – subject to not exceeding air emission limits in real-time. This can be accomplished via process control software and other operations optimization techniques to create optimal manufacturing operations, including unmanned operations, using adaptive control strategies²⁴. An example of the SAIC Process Control Prediction and Optimization tool (demonstration version) is shown on Figure 4.

SOLUTION SET #2 – SOIL STABILIZATION (Batch Process Material Design)

In-situ stabilization of radioactive or heavy metal residue in waste lagoons is an environmental remediation process receiving global attention.²⁶⁻³² The remediation process consists of mixing the polluted residue with a certain amount of grout – typically with a dual³³ or single³⁴ auger – to form a solid product that significantly reduces the mobility of the contaminants.

But the process of mixing the pollutant and grout is not simple. Engineers know the grain size and grout composition. The key properties of the resultant solid mixture are:

- Material strength (UCS – Unconfined Compressive Strength);
- Hydraulic Conductivity: the ability for the material to resist water from migrating through it; and
- Leachability Index: a measure of the leaching of a contaminant of concern should the material be in contact with water.

We reviewed over five hundred relevant references and made inquiries of several industry experts to determine if any solution existed that could predict these three stabilized waste properties using only grain size and grout composition for inputs. We did not find any such solutions.

We chose to model the UCS variable because it is the best single indicator of complete mixing of the grout with the waste lagoon material. If the grout is not well mixed with the waste material — which would be indicated by a very low UCS value — then the mixing would be deemed incomplete and the waste material is neither solidified nor stabilized. In making this comparison, it is important to understand that the UCS measurement is itself, quite noisy. We have estimated the measurement error to be plus or minus 25% of the measurement amount. Thus, evolved solutions that make predictions within 25% of the measured value are good solutions.

The Discipulus GP technique³⁵ was applied to estimate UCS value of the manufactured product, using only grout composition and grain size as inputs. The results for the UCS property are shown on Figure 5. In brief, the Discipulus GP output was able to predict 85% of the variance of the UCS values ($R^2=0.85$). All predictions were within the 25% measurement error discussed above for this parameter. (Although graphs are not included, Discipulus was also applied to the other key output parameters, Leachability Index and Hydraulic Conductivity. Discipulus evolved programs that predict these parameters to within 10% and one order of magnitude, respectively.) The programs evolved by Discipulus for all three parameters make predictions that are well within acceptable margins of error for their intended purpose.

This technique can easily be extended to other batch manufactured materials in many industries.

SOLUTION SET #3 – REMOTE SENSING VIA CONE PENETROMETER TO INFER SUBSURFACE PERCENTAGE OF FINE-GRAINED SEDIMENT (Remote Exploration)

Grain size is a very important property of soils. Grain size affects the soil's ability to transmit water, which is important in water resource planning and subsurface contaminant fate and transport modeling, as well as soil structural properties for foundation engineering and the like. One measure of grain size distribution is the percentage of sediment grains smaller than a specified size.

The conventional method for collecting the grain size distribution in soils is to excavate a soil sample and analyze it in a laboratory using a series of sieves. Alternatively grain-size distribution may be inferred in-situ by a sensing device known as a cone penetrometer (CPT). The CPT consists of a rod with a conical tip and friction sleeve that is pushed forcefully into the ground. During insertion, the penetrometer measures: (1) the resistance of the soil normal to the tip of the CPT; (2) the frictional or tangential stress generated along the sleeve located behind the conical tip; and often (3) the groundwater pressure in soil pores. A function that maps tip resistance, sleeve friction and pore pressure to the percentage of fine-grained soil in the subsurface would allow rapid, cost effective collection of geotechnical data.

The Discipulus GP technique was applied to a data set consisting of known CPT data as well as the corresponding percentage of fine-grained soil. The results are shown on Figure 6. The R^2 value of this relationship is 0.88, which is an improvement over the next best available analysis result of R^2 of 0.81. This improved relationship is more than acceptable for its intended purpose. (Interim results from another CPT data set analysis project show improvement of the predictive relationship for percent fines from $R^2=0.45$ using a linear regression to $R^2=0.69$ for Discipulus GP.)

This technique has wide ranging applications to evaluating remotely collected data to infer other geotechnical properties such as hydraulic conductivity, interpret geophysical surveys such as electromagnetic and ground penetrating radar surveys, identify unseen conditions such as buried structures or ordnance and locate mineral deposits or oil/natural gas reserves.

TESTING GENERALIZATION

The company that developed Discipulus claims that GP in general, and Discipulus, in particular, are not nearly as prone to memorization and overfitting problems as are neural networks and decision trees. There is, in fact, some good published support for the good generalization properties of AIMLearning-based systems.⁷

This section reports the results of tests to date of the generalization capabilities of Discipulus. To date, three separate tests have been run. Two of these tests validate solutions evolved by Discipulus for the problem sets described above on independent, validation data. The third tests the generalization capabilities of Discipulus by presenting it with chaotic time series data, where there is no relationship, to see if Discipulus will find a false relationship.

The results of these three tests of generalization are:

Rotary Kiln Carbon Dioxide Prediction Data

For the Rotary Kiln Incineration and off-gas discharge data sets, the validation work is currently ongoing. As part of those tests, an extensive separate study tested the Discipulus evolved carbon-dioxide emissions results on validation data on which the GP algorithm was not trained. In that study, the Discipulus GP system evolved solutions that fit the CO_2 validation data very well, $R^2 = 0.96$.⁶⁵

UCS Soil Stabilization Data

When we ran the UCS Soil Stabilization data discussed above, we trained Discipulus on data from three soil basins. The best, evolved program was run on data from a fourth, and physically separate, soil basin. The results were within the 25% error measurement on the UCS data discussed above.

Chaotic Time Series Data

In order to further assess whether the Discipulus GP system was fabricating results or memorizing input data, a chaotic times series was constructed in an attempt to deceive Discipulus into predicting an unpredictable relationship. This chaotic series is characterized by the output values wandering randomly about within a range of values. The important point about this chaotic time series is that the information content of the preceding values is not sufficient to predict the next value. In this circumstance, if the GP

technique found a relationship on this chaotic series, the claims of its proprietors regarding memorization and overfitting could be rejected.

The time series used was constructed via a physical process experimental technique discussed in Scientific American. This time series meets the above criteria – to wit, the information content of the values for the any number of preceding time steps is unrelated to the next value in the sequence.³⁶ To gather these data, a cylinder was constructed and filled with mineral oil. Drops of colored water were then metered into the column at varying flow rates. Some of the drops coalesce into larger drops on their way down. Others reach the bottom without combining with other drops. Each time any drop of the colored material reaches the bottom, we recorded how many colored drops had combined on their way down to form that drop. In all, we generated a time series of over eight hundred data points.

We then configured Discipulus to train on a data set configured as follows: (1) The inputs were comprised of eight consecutive values from the drop data; and (2) the target output was the next-in-sequence value of the drop data. Various attempts were tried to trick the technique, including varying the functions that were available for evolving the solution.

The results of the analysis are shown on Figure 7. The genetic programming technique was not fooled by this data set. It evolved a program that was approximately a linear representation of the average value of the data set but did not memorize or fit the noise. Not generating a false relationship is a very important result and provides as much credence to the Discipulus GP technique as do the above three sets of results on the environmental engineering data.

DISCIPULUS INTEGRATION WITH THE SAIC KNOWLEDGE-SLEUTH TOOL-KIT

The SAIC KnowledgeSleuth Tool-Kit is a comprehensive set of programs for integrating data and information to generate the knowledge needed to facilitate optimal decision-making. Discipulus is both a stand-alone application and a component of this tool-kit. The tool-kit consists of three main components, the data acquisition, analysis and knowledge discovery and decision support. The data acquisition component is implemented via an agent-based tool “AgentMiner”. This tool integrates various databases and warehouses, legacy, Intranet, Internet or otherwise, to act coherently as a whole³⁷. The technology is written in JAVA hence can run on any platform and is JDBC and ODBC compliant and can link to virtually any data source. The user then runs a single query from a single station, and all data sources are queried, the results compiled, and displayed or stored in a file. The analysis component consists of a variety of artificial intelligence, machine learning and other techniques that are used to make sense out of the information. Representative techniques include genetic programming, genetic algorithms, analytical neural networks, rule-based classifiers, statistics and other soft computing techniques³⁸⁻⁴⁹. Decision support techniques for optimizing decisions, including stochastic scenarios, are then invoked⁵⁰⁻⁶². Discipulus, and AIMLearning technology in general, is a very effective algorithm, which vastly improves the performance of this tool.

CONCLUSIONS

AIMLearning Genetic Programming, through the software package Discipulus, is ready for immediate, production-level implementation. This is based on the following findings:

- AIMLearning based systems were able to solve real problems that heretofore were difficult, if not impossible, to solve with conventional methods. This is a result of both the genetic programming technique and its implementation at the machine code level.
- The AIMLearning implementation in machine code is, in fact, sixty to two hundred times faster than other methods of Genetic Programming.
- AIMLearning solutions are packaged as C/C++ code or assembly language, which translate directly into process prediction software applications, embedded applications, and control algorithms.

- AIMLearning solutions appear to have excellent generalization properties, an important advantage over ANN and decision tree derived solutions.

In conclusion, this new technology is a valuable addition to the SAIC Knowledge Sleuth Tool-Kit.

ACKNOWLEDGEMENTS

Science Applications International Corporation funded the preceding work. Specific support and encouragement was provided by Dr. Clinton W. Kelly, III, Senior Vice President and Director of SAIC Internal Research and Development, and Messers. Joseph W. Craver -Group Manager, Environmental and Engineering Management Group, Janardan J. Patel -EEMG Division Manager, and Fred Zafran -Vice President, Enterprise Management Information Systems. Dr. Jenifer McCormack and her staff did an excellent job on extending the InfoSleuth capabilities into the new SAIC *AgentMiner* tool. The Department of Energy and the Westinghouse Savannah River Corporation, specifically Dr. Gregory Flach and Frank Syms, are gratefully acknowledged for providing the input data sets used in some of the work. Christopher R. Wellington -Ad Fontes Academy, Centreville, Virginia conducted the chaotic drop experiment. Edward Dilkes supplied many comments and insights, particularly on the dynamic process simulation (CIF) example. All computations were performed by, and responsibility for their accuracy lies with, Larry M. Deschaine, PE.

REFERENCES

1. Nordin, J.P., (1999), *Evolutionary Program Induction of Binary Machine Code*, ISBN 3-931546-07-1, Krehl Verlag, 290 pp.
2. Nordin, J.P. (1994). A Compiling Genetic Programming System that Directly Manipulates the Machine Code. In: *Advances in Genetic Programming*, K. Kinnear, Jr. (ed.), Cambridge, MA , USA, MIT Press.
3. Nordin, J.P. and Banzhaf, W. (1995). Evolving Turing Complete Programs for a Register Machine with Self-Modifying Code. In: *Proceedings of the Sixth International Conference of Genetic Algorithms*, Pittsburgh, Penn., USA, Morgan Kaufmann Publishers
4. Nordin, J.P. and Banzhaf, W. (1995). Complexity Compression and Evolution. In: *Proceedings of the Sixth International Conference of Genetic Algorithms*, Pittsburgh, Penn., USA, Morgan Kaufmann Publishers
5. Nordin, J.P., Francone, F. and Banzhaf, W. (1995) Explicitly Defined Introns in Genetic Programming. In: *Proceeding of The GP workshop at 12th International Conference on Machine Learning: From Theory to Real-World Applications*, Rosca, J.P. (Ed.) Tahoe City, USA.
6. Nordin, J.P. and Banzhaf, W. (1996) Controlling an Autonomous Robot with Genetic Programming. In: *Proceedings of 1996 AAAI fall symposium on Genetic Programming*, Cambridge, USA.
7. Francone, Nordin, J.P. and Banzhaf W. (1996) Benchmarking The Generalization Capabilities of a Compiling Genetic Programming System Using Sparse Data Sets. In: *Proceedings of The First International Conference on Genetic Programming*, Stanford, USA.
8. Nordin, J.P. and Banzhaf W. (1996) Programmatic Compression of Images and Sound. In: *Proceeding of The First International Conference on Genetic Programming*, Stanford, USA.
9. Nordin, J.P., Francone, F. and Banzhaf, W. (1996) Explicitly Defined Introns in Genetic Programming. In *Advances in Genetic Programming II*, eds. Kim Kinnear, Peter Angeline, MIT Press USA.
10. Nordin, J.P. (1996) A Guide and Tutorial on Machine Code Genetic Programming. Tutorial Notes at: *The First International Conference on Genetic Programming*, Stanford, USA.
11. Banzhaf W., Francone, F. and Nordin P J. (1996) The Effect of Extensive Use of the Mutation Operator on Generalization in Genetic Programming using Sparse Data Sets. In: *Proceedings of Parallel Problem Solving From Nature IV. Proceedings of the International Conference on Evolutionary Computation*. Springer-Verlag.
12. Banzhaf, W. Nordin, P. J. and Olmer, M. (1997) Generating adaptive Behavior for a Real Robot using Function Regression within Genetic Programming. In: *Proceedings of the Second Annual Conference on Genetic Programming*, Stanford USA.
13. Nordin, P., Banzhaf, W. (1997) Genetic Reasoning Evolving Proofs with Genetic Programming. In: *Proceedings of the Second Annual Conference on Genetic Programming*, Stanford USA.

14. Banzhaf, W., Nordin, P. J., Keller, R., Francone, F. (1998) *Genetic Programming – An Introduction. On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann Publishers, Inc. 470 pp.
15. Fausett, L. (1994) *Fundamentals of Neural Networks- Architectures, Algorithms, and Applications*. Prentice Hall Publishers, 461 pp.
16. Skapura, D. M. (1996) *Building Neural Networks*, Addison Wesley Publishers, 1996, 286 pp.
17. Swingler, K. (1996) *Applying Neural Networks – A Practical Guide*, Academic Press, 303pp.
18. Bigus, J. P. (1996) *Data Mining with Neural Networks – Solving Business Problems from Application Development to Decision Support*, McGraw Hill, 220 pp.
19. Koza, Bennett, Andre, & Keane, (1999) *GENETIC PROGRAMMING III – Darwinian Invention and Problem Solving*, Morgan Kaufmann Publishers, Inc. 1154 pp.
20. Francone, F., (1998-2000) *Discipulus Owner’s Manual* and *Discipulus Tutorials*, Register Machine Learning Technologies, Inc.
21. Register Machine Learning Technologies, 11757 Ken Caryl Ave. #F-PMB512. Littleton CO USA 80127, web page: <http://www.aimlearning.com>
22. Murrill, P.W., (2000) *Fundamentals of Process Control Theory, 3rd Edition*, Instrument Society of America; ISBN: 155617683X, 300pp.
23. Laplante, Phillip A. (1999) *Real-Time Systems Design and Analysis, An Engineers Handbook, 2nd Edition*, IEEE Press, 361 pp.
24. Francone, F.D., Nordin, P.J., Banzhaf, W., Dilkes, E., Deschaine, L.M. (2000) AIM Learning™ Adaptive, Real-Time, Control Technologies. In: *Society for Computer Simulation’s Advanced Simulation Technology Conference*, Washington, DC, USA April 2000, ISBN: 1-56555-199-0, pp. 9-12.
25. Fausett, L. V., A Neural Network Approach to Modeling a Waste Incinerator Facility, In: *Society for Computer Simulation’s Advanced Simulation Technology Conference*, Washington, DC, USA April 2000 ISBN: 1-56555-199-0, pp. 19-23.
26. Stegemann, J.A. and Buenfeld, N.R. (2000) Synthesis of solidification experience for synthetic wastes, accepted for presentation at WASCON 2000, May 31-June 2, 1999, to be published as Science and Engineering of Recycling for Environmental Protection, in the series *Studies in Environmental Science*, Elsevier Science B.V., Amsterdam, 2000.
27. Stegemann, J.A. and Buenfeld, N.R. (1999), *Predicting Interactions of Cement with Waste, Beneficial Use of By-Product Materials in Construction Applications*, November 15-16, 1999, Albany, New York., W. Chesner, Ed.
28. Stegemann, J.A., Butcher, E.J., Ouki, S., Irabien, A., de Miguel, R., Johnston, P., Sassaroli, G. and Sirini, P. (1999), Predicting the Effects of Impurities in Cement - A Workshop for End-Users, *Waste Stabilisation And Environment*, 13-16 April, 1999, Lyon Villeurbanne, J. Mehu, G. Keck and A. Navarro, Eds., ISBN 2-905015-40-3, Société Alpine de Publications, Grenoble, pp. 225-254,1999.
29. Stegemann, J.A. and Buenfeld, N.R. (in Progress) *Neural Network Analysis for Prediction of Interactions in Cement/Waste Systems: Introduction*, pp. 224-231.
30. Stegemann, J.A. and Buenfeld, N.R. (In Progress) *Neural Network Analysis for Prediction of Interactions in Cement/Waste Systems: Task 3. Neural Network Analysis*, pp. 243-246.
31. Hills, C.D., Stegemann, J.A. and Buenfeld, N.R. (1997) Investigating Waste/Binder Interactions by Neural Network Analysis, *Waste Materials in Construction: Putting Theory in Practice*, *Studies in Environmental Science* 71, J. Goumans, J. Senden and H. van der Sloot, Eds., ISBN 0-444-82771-4, Elsevier Science B.V., Amsterdam, pp. 421-430, 1997.
32. Stegemann, J. (1998) *Neural Network Analysis for Prediction of Interactions in Cement/Waste Systems (NNAPICS)*, URL: <http://concrete-www.cv.ic.ac.uk/iscowaa/nnapics/intro.html> April, 1998.
33. In-Situ Fixation, Inc. P.O. Box 516 Chandler, AZ 85244, Phone: (480) 821-0409 Fax: (480) 786-3184 web page <http://www.insitufixation.com>
34. Geo-Con, Inc. 4075 Monroeville Blvd, Corporate One, Building II, Suite 400, Monroeville, PA 15146, Phone: (412) 856-7700 Fax: (412) 373-3357; web page <http://www.geocon.net>
35. Deschaine, L. M., Zafran, F. A., Patel, J. J., Amick, D., Pettit, R., SAIC, Francone, F. D., Nordin, P., Dilkes, E., RMLT, and Fausett, L. V. (2000) Solving the Unsolved-Using Machine Learning to Model a Complex Production Process – Case Examples Applying Three Machine Learning Techniques, *Society for Computer Simulation’s Advanced Simulation Technology Conference*, Washington, DC, USA April 2000 ISBN: 1-56555-199-0, pp. 3-8.

36. Scientific American (1999) Drop Experiment to Demonstrate a Chaotic Time Series, November Magazine.
37. Deschaine, L. M., Brice, R. S., Nodine, M. H., Use of InfoSleuth to Coordinate Information Acquisition and Analysis in Complex Applications. *Society for Computer Simulation's Advanced Simulation Technology Conference*, ISBN 1-56555-199-0, pp. 13-18. Washington, DC, USA April 2000.
38. Weiss, S. M., Indurkha, N. (1998) *Predictive Data Mining – A Practical Guide*, Morgan Kaufmann Publishers, 228 pages.
39. Fidler, N. V. (1991) *An Artificial Intelligence Technique for Information and Fact Retrieval – An Application in Medical Knowledge Processing*, The MIT Press, Cambridge, MA, 155 pp.
40. Eberhart, R., Simpson, P., Dobbins, R., (1996) *Computational Intelligence PC Tools*, Harcourt, Brace & Company – Academic Press, 464 pages.
41. Sinha, N., K., Madan, M., G. (2000), *Soft Computing & Intelligent Systems – Theory and Applications*. Harcourt, Brace & Company – Academic Press, 639 pages.
42. Spector, L., Langdon, W., B., O'Reilly, U., Angeline, P.J. (1999) *Advances in Genetic Programming – Volume 3*, MIT Press, 476 pp.
43. Mann, K. F., Tang, K. S., Kwong, S. (1999) *Genetic Algorithms – Concepts and Designs*, Springer-Verlag, 344 pp.
44. Rao, V., B., Roa, H., V. (1995) *C++ Neural Networks & Fuzzy Logic, 2nd Ed.* MIS:Press, 551 pp.
45. Giarratano and Riley (1998) *Expert Systems – Principals and Programming, 3rd Ed.*, International Thomson Publishing, 597 pp.
46. Thomsen, E. (1997), *OLAP Solutions – Building Multidimensional Information Systems*, Wiley, 576 pp.
47. Kimball, R. (1996), *The Data Warehouse Toolkit, Pratical Techniques for Building Dimensional Data Warehouses*, Wiley, 374 pp.
48. Berson, A., Smith, S., J. (1997), *Data Warehousing, Data Mining, & OLAP*, McGraw Hill, 612 pp.
49. Quinlan, R., J., (1993) *C4.5 – Programs for Machine Learning*, Morgan Kaufmann Publishers, 302 pp. Note: This algorithm extended to C5 – see RuleQuest.com for information as publication not available.
50. Winston, W., L.(1996) *Simulation Modeling using @Risk*, Duxbury – International Thomson Publishing, 230 pp.
51. Winston, W. L. (1996) *Management Science: Applications and Spreadsheet Modeling*, ISBN 0-534-21774-5, Duxbury – International Thomson Publishing.
52. Winston, W. (1994) *Operations Research – Applications and Algorithms*, Duxbury – International Thomson Publishing, 1318 pp.
53. Winston, W., (1999) *Decision Making Under Uncertainty*, Palisade Corporation, 244 pp.
54. Winston, W. (1998) *Financial Models Using Simulation and Optimization*, Palisade Corporation, 499 pp.
55. Ragsdale, C., T. (1998), *Spreadsheet Modeling and Decision Analysis 2nd Ed.*, Southwestern College Publishing – ITP, 742 pp.
56. Kirkwood, C. W. (1997) *Strategic Decision Making – Multi-Objective Decision Analysis with Spreadsheets*, Duxbury – ITP, 345 pp.
57. Kaplan, R., S., Norton, D., P. (1996) *The Balanced Scorecard*, Harvard Business School Press, 322 pp.
58. Saaty, T. L. (1996) *Decision Making for Leaders*, Vol. II., RWS Publications, 315 pages.
59. Fornam, E. H. (2000), *Decision Making By Objectives*, George Washington University, (In Progress).
60. Saaty, T., L (1991) *Prediction, Projection, and Forecasting*, Kluwer Academic Publishers, 251 pp.
61. Saaty, T., L. (1996) *The Analytic Network Process – Decision Making with Dependence and Feedback*, RWS Publications, 370 pp.
62. Dyer, R. F., Forman, E. H. (1991) *An Analytic Approach to Marketing Decisions*, Prentice Hall, 434 pp.
63. Koza, J.R., (1992) *Genetic Programming: On the Programming of Computers by Natural Selection*. MIT Press, Cambridge, MA.
64. Fukunaga, A., Stechert, A. Mutz, D. Jet Propulsion Laboratories, California Institute of Technology Pasadena, CA. (1998). A Genome Compiler for High Performance Genetic Programming, in *Proceedings of the Third Annual Genetic Programming Conference*, pp. 86-94. Morgan Kaufman Publishers, 1998.
65. Deschaine, L.M. (2000) *Using Genetic Programming to Develop a C/C++ Simulation Model of a Waste Incinerator*, Science Applications International Corp., Draft Technical Report.

Figure 1. Waste Incineration: Results of Genetic Programming Simulations - Rotary Kiln Incineration Temperature. Average Error = 2.0%, Maximum Error = 8.6% (Series 2 - the squares - are the GP Predictions)

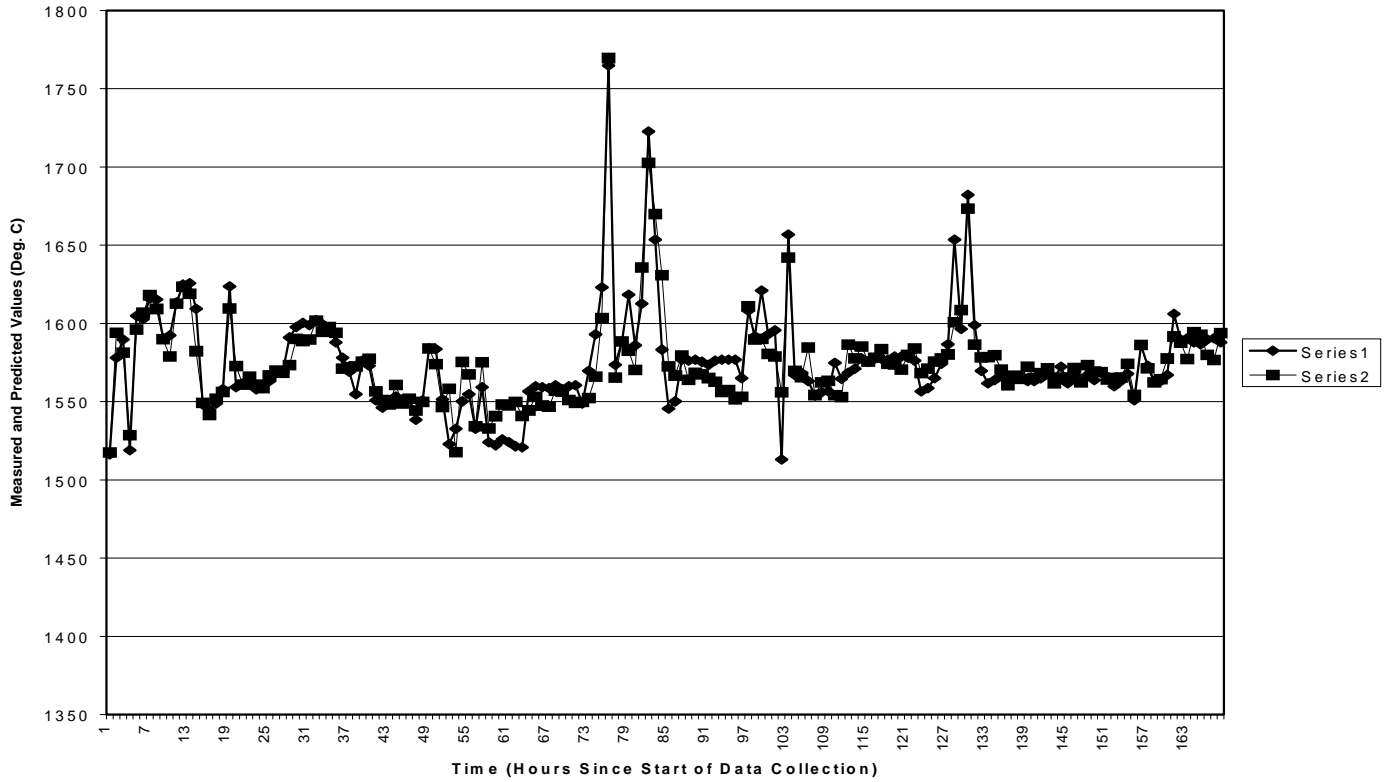
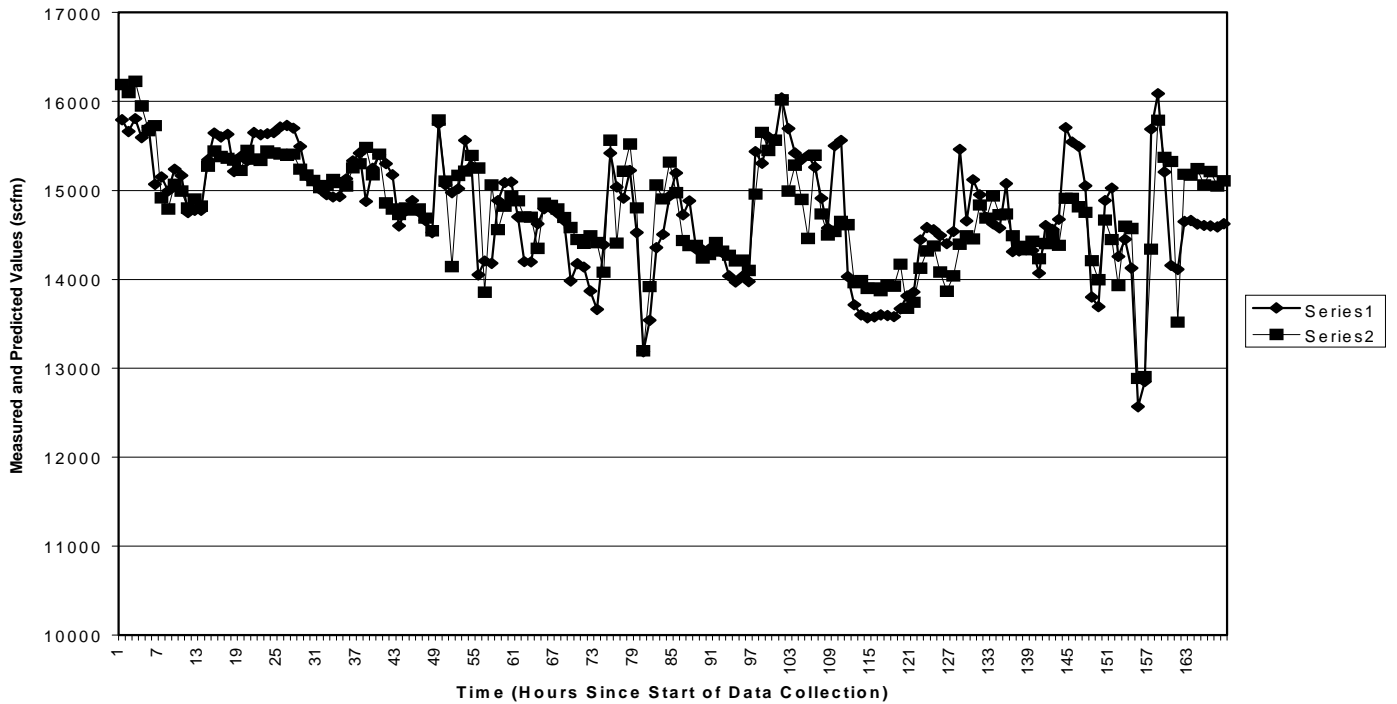
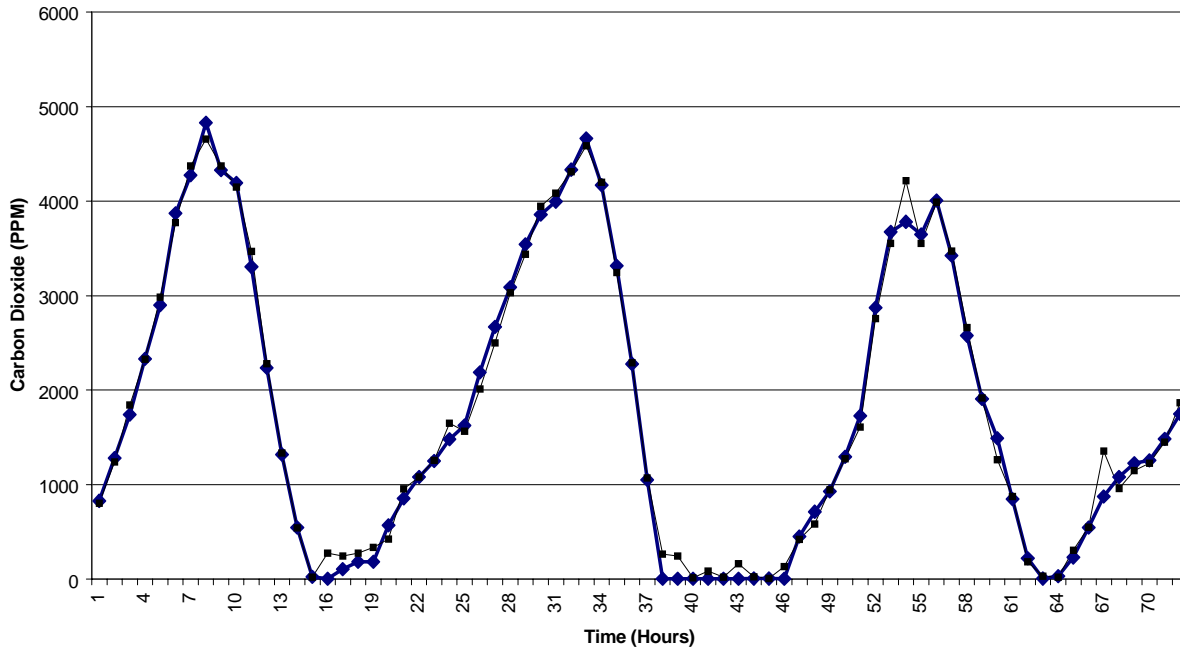


Figure 2. Waste Incineration: Results of Genetic Programming Simulations - Off-Gas Duct Flow Rate. Average Error = 0.6%, Maximum Error = 3.2% (Series 2 - the squares - are the GP Predictions)



**Figure 3. Full-Scale Waste Incinerator
Carbon Dioxide Emissions
Actual (heavy line) vs. Predicted (light line) Values
Genetic Programming Valadation Mass Balance = 1.2%, R² = 0.993**



SAIC-USC AIKEN INCINERATOR PROCESS CONTROLLER (Demo Version)

INPUT INFORMATION

Rotary Kiln Incinerator (lbs/hr)		Secondary Combustion Chamber (lbs/hr)		Input Data Sources <input type="checkbox"/> Plant Floor (real-time) <input type="checkbox"/> Plant Historian (off-line) <input checked="" type="checkbox"/> Screen (Manual Input) <input type="checkbox"/> Company Intranet <input type="checkbox"/> W/W Internet (Pricing Only) <input type="checkbox"/> Stored File Reporting Options <input type="checkbox"/> MS Office Applications <input type="checkbox"/> ASCII File <input type="checkbox"/> Env. Mgt. Info. System
Fuel Oil (flow)	99.844	Fuel Oil (flow)	357.03	
Liquid Waste (flow)	228.96	Fuel Oil (air flow)	5734.4	
Solids (air flow)	960.94	Steam (flow)	10050	
Solids (flow)	389.84	Cost Information (\$/unit)		
Solids (average flow)	473.44	Fuel Oil (gal.)	1.29	
Fuel Oil (air flow)	1587.5	Steam	0	
Liquid Waste (air flow)	4037.5			
Aqueous Waste (flow)	291.15			

PROCESS OUTPUTS

Rotary Kiln Incinerator		Incinerator Off-Gas Metrics	
Temperature (C)	1608.1	CO2 #1 (ppm)	2.03
Secondary Combustion Chamber		CO2 #2 (ppm)	2.34
Temperature (C)	1804.7	CO2 #3 (ppm)	2.5
Oxygen (%)	21.178	CO2 #4 (ppm)	.78
CO2 (ppm)	829.43	Oxygen #1 (%)	11.04
		Oxygen #2 (%)	11.06
		Duct Flow #1 (scfm)	15438
		Duct Flow #1 (scfm)	13575

Figure 4: Process Controller User Interface (Demonstration Version)

Figure 5. Genetic Programming Results of Soil Stabilization Mix Design - Unconfined Compressive Strength at 28 days
(Series 2 - the squares - are the GP Predictions)

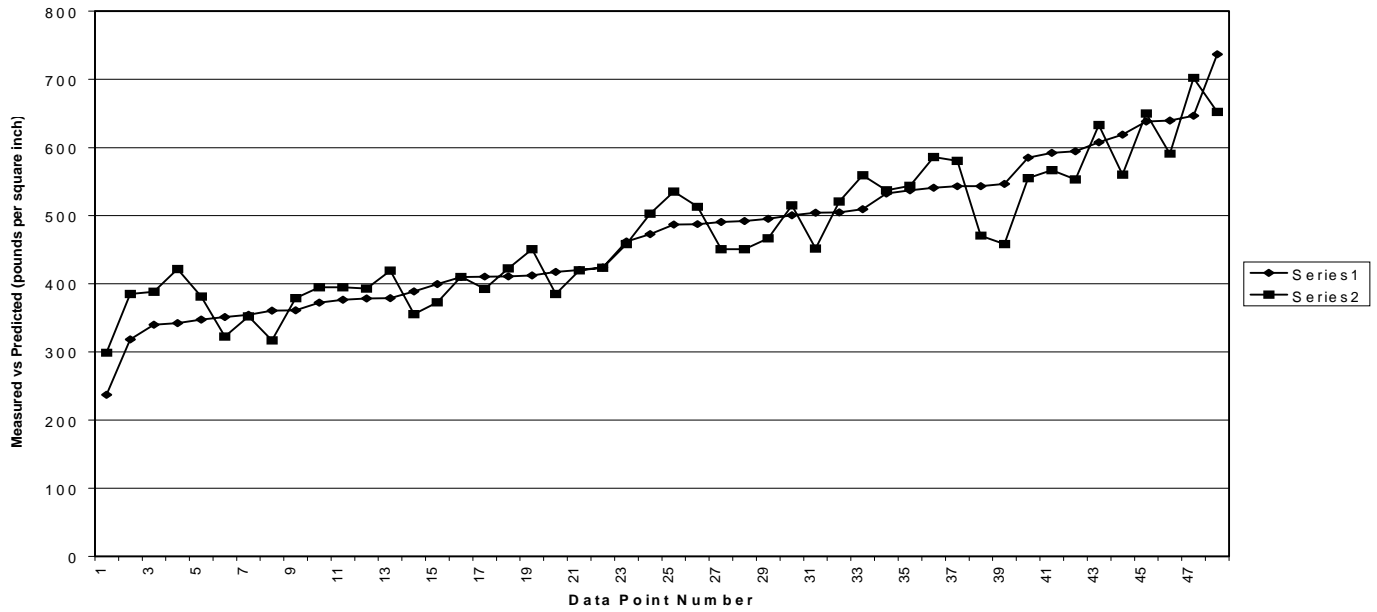


Figure 6. Genetic Programming Results of CPT Data Used to Predict Geotechnical Properties
(Series 2 - the squares - are the GP Predictions)

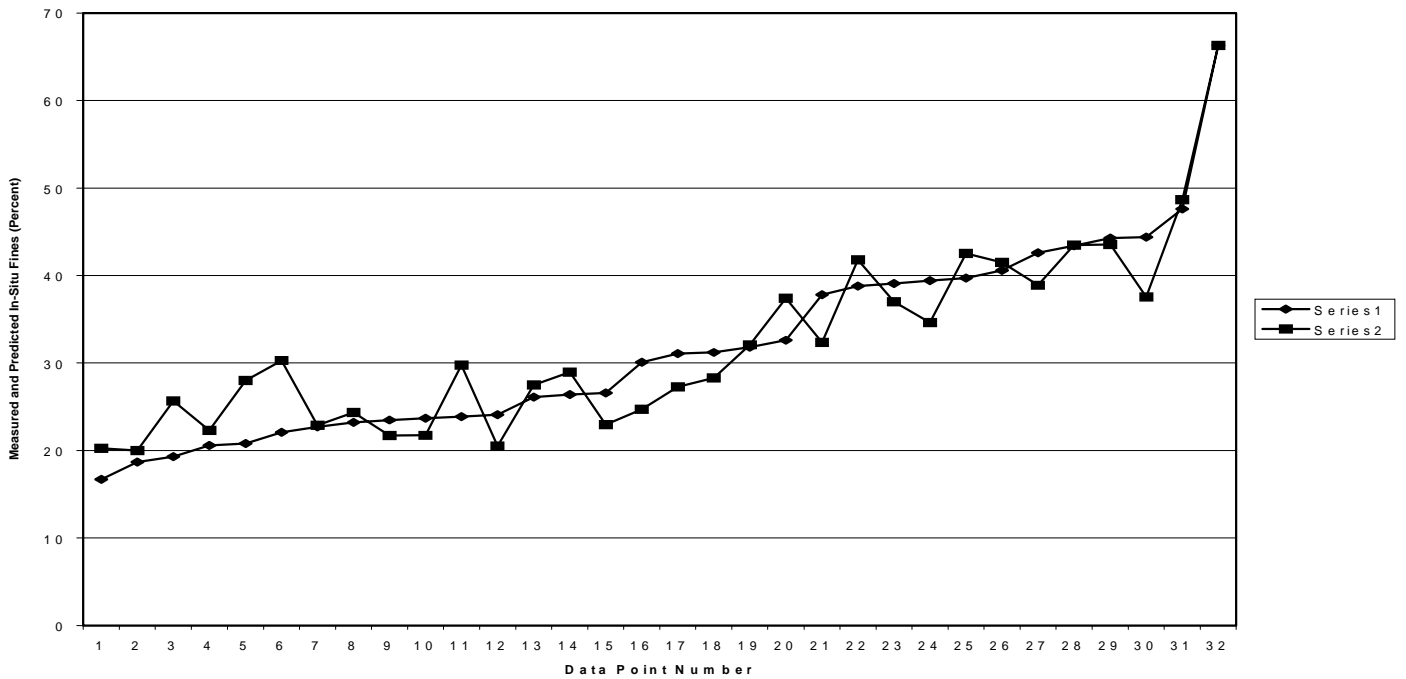


Figure 7. Results of Attempt to Decieve Genetic Programming via Chaotic Series
GP Technique was not Deceived
(The squares - are the GP Predictions)

