Dimensionally Aware Genetic Programming

Maarten Keijzer
Danish Hydraulic Institute
Agern Allé 5, DK-2970
Hørsholm, Denmark

Vladan Babovic
Danish Hydraulic Institute
Agern Allé 5, DK-2970
Hørsholm, Denmark

Abstract

Physical measurements are generally accompanied by their units of measurement. This contribution introduces an extension of genetic programming that exploits the information captured in the units of measurement and compares it against standard methods of genetic programming. The motivations for the development of this dimensionally-aware GP are twofold: to enhance the search efficiency by utilising the knowledge contained in the dimension information and to enhance the interpretability of the produced formulae. The performance of GP is examined on a number of experiments and the results are reported for four variants of GP.

1 INTRODUCTION

In scientific endeavours, data represents carefully collected observations about the particular phenomena that are under study. Physical observations are usually accompanied by their units of measurement. However, the traditional methods usually eliminate this information through the introduction of dimensionless ratios (well known examples are the Mach number and the Reynolds number). Once the dimensionless numbers are used instead of the original dimensional values, the problem of dimensional correctness is conveniently avoided, as all analysed quantities are dimension-free. It is also argued that dimensionless ratios 'collapse' the original dimensional search space, making it more compact, thus resulting in a more effective behaviour of algorithms that fit models to data. At the same time, the information contained in the units of measurement can be ignored entirely.

The primary questions addressed in this contribution are: *"What must one know a priori about an unknown functional dependency in order to estimate it on the basis of observations? Is it possible to extend the present GP paradigm, without making it too strong, and to enhance both the interpretability of induced relationships and the efficiency of the evolutionary search process? What can the notion of dimensionality contribute to answering these questions?"*

1.1 INTRODUCING UNITS OF MEASUREMENT: A SEARCH PERSPECTIVE

Search has always been an integral part of machine intelligence. In principle, all problem-solving strategies in Artificial Intelligence (AI) can be divided into *strong* and *weak* search strategies. Weak search methods *do not* make any assumptions about the domain they are traversing. Strong methods *do*. Consequently, since the strong methods utilise the auxiliary information in a form of explicit knowledge about the domain they are trying to solve, they usually outperform the weaker methods. However, strong methods can be successfully applied only to those classes of problems where the assumptions can be justified and conveniently introduced. Since weak methods in principle assume nothing, they have, in principle, an almost universal applicability.

In a context of scientific discovery, one usually takes advantage of already available scientific knowledge about the domain in question. The incorporation of such knowledge into a process of search for new, more accurate, insightful relationships, makes the search algorithm stronger and consequently deflates its universal applicability. Some scientific disciplines even have their own 'recipes' that rely on so much domain-dependent knowledge that they invalidate the application of these recipes in other areas.

Genetic Programming (GP) is a weak search algorithm operating on parse trees of functions and terminals. In its native form, GP lends itself quite naturally to the process of induction of mathematical models based on

observations: GP is an efficient search algorithm that need not assume the functional form of the underlying relationship. Given an appropriate set of basic functions, GP discovers a (sometimes very surprising) mathematical model that approximates the data well. At the same time, GP-induced models come in a symbolic form that can be interpreted by scientists (see for example, Babovic, 1996).

Unfortunately the application of standard GP in a process of scientific discovery does not guarantee satisfactory results. In certain cases, despite their remarkable accuracy, GP-induced relationships are too complicated and provide little new insight into the process that generated the data. One may argue that GP, in such situations, blindly fits parse trees to the data, in almost the same way as that in which Taylor or Fourier series are expanded to approximate observations. It can be argued that GP then results in a model with accurate syntax, but with meaningless semantics. In these cases, the dimensions of the induced formulae often do not fit, pointing to the physical uselessness of the induced relationships.

Standard GP is ignorant of the dimensionality of its terminals and can safely be applied to problems composed of dimensionless numbers only. Given the symbolic nature of GP and its ability to manipulate the structure of functional relationships, it seems strange that information contained in units of measurement has so far not been used as an aid in search process. After all, the dimensional correctness as used in science acts as a syntactic constraint on *any* formula it induces. It is therefore expected that the introduction of dimensions in the GP paradigm might result in improved search efficiency.

2 EXTENDING GP

GP, as an instance of the evolutionary algorithm family, iteratively applies variation and selection on a population of evolving entities. Standard variation operators in genetic programming are subtree mutation (replace a randomly-chosen subtree with a randomly generated subtree) and subtree crossover (replace a randomly chosen subtree from a formula with a randomly chosen subtree from another formula). In order to accommodate the additional information available through units of measurement, the following extensions of standard GP are proposed.

2.1 INTRODUCTION OF UNITS IN GP

In the dimension-augmented setup, every node in the tree maintains a description of the units of the measurement it uses. These units are stored as arrays of real-valued exponents of the corresponding dimensions. In the present set of experiments only the dimensions of length, time and mass (LTM) are used, but the setup may be trivially extended to include all other SI-dimensions (amount of substance, electric current, thermodynamic temperature and luminous intensity). Square brackets are used to designate units, for example [1, -2, 0] corresponds to a dimension of acceleration ($L^1T^{-2}M^0$). Similarly, [0,0,0] defines a dimensionless quantity.

2.2 DEFINITION OF THE TERMINAL SET

The definition of the terminal set is straightforward, in that variables and constants are accompanied with the exponents of their respective units of measurement:

$$T = \{ v_1[x_{v1},y_{v1},z_{v1}], \ldots, v_n[x_{vn},y_{vn},z_{vn}], \\ c_1[x_{c1},y_{c1},z_{c1}], \ldots, c_m[x_{cm},y_{cm},z_{cm}] \} \quad (1)$$

Where *v* designates a variable, *c* a constant and [x,y,z] the corresponding array containing the units of measurement. For example, $v_1[0,0,1]$ designates a variable – v_1 – with a dimension of mass. User-defined constants can be defined along with their dimensions, such as 9.81[1, -2,0] defining the earth's gravitational acceleration. Randomly generated constants are allowed only as dimensionless quantities ([0,0,0]). There is a definitive reason for allowing random numbers to be dimensionless only. Should random constants with random dimensions be allowed, GP would have an easy way to correct the dimensions by introducing transformation from one arbitrary unit of measurement to another. Some form of pressure should be applied on the application of unit transformation. This issue is addressed during the experimentation.

2.3 DEFINITION OF THE FUNCTION SET

Application of arithmetic functions on dimension-augmented terminals violates the closure property for these functions (Koza, 1992). For example, adding meters to seconds renders a dimensionally incorrect result of the operation. Therefore, the definition of arithmetic operators is augmented to: (1) specify the transformation of units of measurement, (2) accommodate units of measurement-related constraints on the application of functions and (3) introduce additional functions that repair trees and provide the benefits of closure.

Table 1 summarises the effects of the application of functions on units of measurement and specifies constraints on the applications of functions. For example, exponentiation of a value can only take place when the operand is dimensionless, in which case the result of the operation is also a dimensionless value. Similarly, addition and subtraction are constrained so that their operands must have the same dimensions. Multiplication and division combine the exponents by adding and subtracting the dimension exponents respectively. The standard `Power` function can be applied to dimensionless values only, whereas `PowScalar` can be applied to dimensional operands, affecting their dimensions correspondingly. Other functions can be defined in similar ways.

Table 1 Effects and constraints that units of measurement impose on function set

Function	Operand Dimensionality	Result
Exponentiation:	[0,0,0]	[0,0,0]
Logarithm:	[0,0,0]	[0,0,0]
Square Root:	[x,y,z]	[x/2, y/2, z/2]
Addition:	[x,y,z], [x,y,z]	[x,y,z]
Subtraction:	[x,y,z], [x,y,z]	[x,y,z]
Multiplication:	[x,y,z], [u,v,w]	[x + u ,y + v, z + w]
Division:	[x,y,z], [u,v,w]	[x - u ,y – v, z - w]
Power:	[0,0,0], [0,0,0]	[0,0,0]
PowScalar (c):	[x,y,z]	[x*c, y*c, z*c]
If less than zero:	[0,0,0], [x,y,z], [x,y,z]	[x,y,z]

2.3.1 DimTransform

As mentioned earlier, an additional function should be introduced in order to guarantee closure:

Function	Operand Dimensionality	Result
`DimTransform` c[x,y,z]:	[u,v,w]	[x + u ,y + v, z + w]

(2)

This transformation operator multiplies its operands with the constant value *c* and also affects dimensions through applying values for *x*, *y* and *z* as indicated in (2). `DimTransform` can be used to resolve dimensional violations that will inevitably arise when using standard-GP-style randomized crossover on formulae with dimensional variables. For example, when meters are added to seconds, the second operand can be transformed into meters by wrapping it with a transformation of magnitude 1.0 and unit description [1,-1,0]. `DimTransform` can therefore transform a quantity expressed in one unit into a quantity expressed in a completely different unit.

At first sight, it appears that the application of `DimTransform` eliminates all the problems related to dimensions. It is evident, however, that the application of `DimTransform` can 'fix' any dimensionality-related problem by introducing physically meaningless transformations into evolving formulae.

Initialisation, crossover and mutation impose heavy demands on the use of this transformation. Whenever a constraint violation occurs (for instance after an insertion of a subtree), a `DimTransform` function which solves this violation is inserted. At present, this is done in an arbitrary manner *i.e.* no attempt is made to find the optimal sequence of transformations for a given formula.

Therefore, the application of `DimTransform` does not always contribute to an interpretable solution to the problem. To control its application an additional objective is introduced (see section 2.4).

The `DimTransform`-based mechanism is not a form of strongly typed GP. A strongly typed GP would initialise and keep all expressions dimensionally correct throughout evolutionary process. For ill-posed and incomplete problems, a strongly typed GP would fail even at initialisation. The current approach allows dimensionally incorrect solutions, but introduces evolutionary pressure on dimensional incorrectness.

2.4 DEFINITION OF GOODNESS-OF-DIMENSION CALCULATION

Apart from the usual *goodness-of-fit* statistic, a second objective — *goodness-of-dimension* — is introduced.As stated earlier, the application of the `DimTransform` operator can result in an arbitrary transformation of units. This is not necessarily a desirable behaviour, as multiple occurrences of this operator do not enhance the interpretability of the resulting formulae. When no constraints are placed on the frequency of `DimTransform`, the approach reduces to a standard GP with the added 'feature' that the resulting formulae grow proportionately to the number of unit violations.

One of the approaches used to reduce the number of applications of `DimTransform` takes full advantage of the explicit representation of the dimensions as a vector of real valued exponents. The distance of an expression from a dimensionally correct formulation can be calculated as the number of required transformations, *i.e.*

$$\textit{Goodness-of-Dimension} = \sum |x_i| + |y_i| + |z_i| \qquad (3)$$

where the subscript *i* ranges over all applications of the `DimTransform` operator in the formula, whereas *x*, *y* and *z* are the components of the dimension vector. The goodness-of-dimension acts as an effective metric of distance from desired dimensions and can be treated as an additional measure of fitness. Goodness-of-dimension can be combined with the goodness-of-fit statistic in a multi-objective optimization fashion (see 3.2).

2.5 DIMENSION-BASED BROOD SELECTION

Brood selection (Tackett, 1994) is a technique where the parents produce a large number of offspring. These offspring are evaluated against a 'cheaper' fitness function (often referred to as the culling function). The best of these offspring are moved to the next generation.

The evaluation of a large collection of the offspring by a culling function improves the overall performance because little effort is wasted on the evaluation of bad offspring on the expensive complete fitness function.

The culling function used in dimensionally-aware GP is the goodness-of-dimension of the formula. This evaluation is very cheap as it can be calculated independently of the training set and it requires a single pass through the parse tree.

The present implementation reads as follows: two parents are chosen for crossover; they produce *m* offspring by repeated application of the random sub-tree crossover operation; constraint violations are corrected for dimensions in the manner outlined in 2.3.1 DimTransform; and, finally, the best among the *m* offspring with respect to goodness-of-dimension are added to the intermediate population.

Therefore, this operation can be best understood as a unit-informed crossover. Although the use of a culling function equates to selection, it is applied immediately after individual crossovers, thus modifying the results of crossover to produce formulae that are dimensionally more correct.

3 EXPERIMENTS

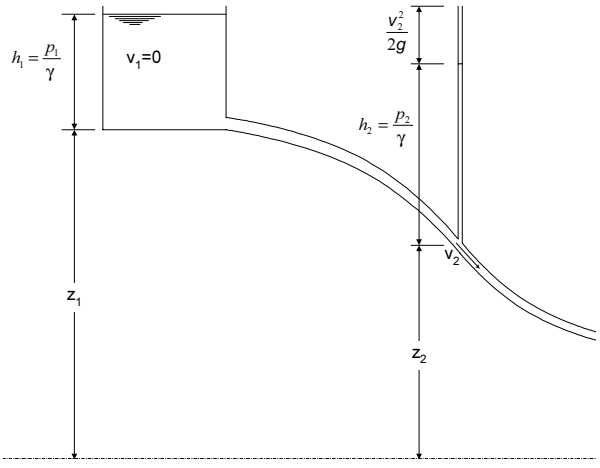

Figure 1: Definition sketch for the energy conservation law for steady, one-dimensional fluid flow

Dimensionally-aware GP can best be tested in a fully controlled experimental setup, that is, one for which a unique solution is known to exist. Also, the experiments should involve non-trivial units of measurement.

Therefore, a choice has been made to generate data using the following scientific law.

3.1 THE ENERGY CONSERVATION LAW FOR STEADY, ONE-DIMENSIONAL FLUID FLOW

Some of the most important principles of modern fluid dynamics are conservation laws. The conservation laws for the flow of fluids state that mass must be conserved and that the impulses must be balanced by changes in momenta. The conservation of energy is another such law formulated by Daniel Bernoulli in early XVIII century. The *Bernoulli equation* expresses the constancy of the sum of kinetic energy, potential energy and work done per unit mass and can be expressed in a simplified form as:

$$E = z_1 + \frac{p_1}{\gamma} + \frac{v_1^2}{2g} = z_2 + \frac{p_2}{\gamma} + \frac{v_2^2}{2g} = const. \quad (4)$$

where
z denotes distance above a certain energy datum,
p denotes pressure,
v denotes velocity,
g denotes the Earth's gravitational acceleration [g=9.81 m^2/s], and
γ denotes the specific gravity of a fluid [for water γ=9810 N/m^3].

Formula (4) is a simplified formulation that ignores energy losses due to friction and local losses induced by sudden changes in a cross-section of the conduit. In accordance with tradition in hydraulics, energy in formulation (4) is expressed in implicit potential energy terms, as meters of a water column, rather than in more conventional energy units.

3.2 EXPERIMENTAL SETUP

Four different versions of evolutionary algorithm have been tested and reported:

- standard GP,
- GP with brood selection,
- multi-objective GP, and
- multi-objective GP with brood selection.

These were designed so that the investigation of the most appropriate way to inject and control knowledge about the units of measurement is conducted. The standard method does not use any dimension information. The brood selection method uses knowledge of dimension within the variation component only. The multi-objective approach is dimensionally-aware only within its selection component. The multi-objective approach combined with dimensionally based brood selection utilises dimensions in both the selection and variation components. Basically, the question addressed is whether most benefit is gained through variation, selection or a combination of the two.

Table 2 Overview of the evolutionary algorithm setup

Parameter	Value
Objective	Find the Bernoulli Equation of Energy Conservation.
μ	250
λ	500
Tournament size	3
Brood size	2 (culling function on unit error)
Crossover probability	1.0
Mutation probability	0.05
Crossover method	Random subtree crossover
Objective Functions	RMSE and/or unit error and fitness sharing
Function set	{+, *, %, -} (% denotes protected division)
Terminal set	Variables: z, v, p; Constants: g, ,γ random constants
Maximum size at initialisation	15
Maximum size	41
Probability of selecting a constant vs. a variable	0.05
Constant mutation probability	0.05

3.2.1 Evolutionary algorithm

Present experiments use a $(\mu + \lambda)$ type of evolutionary process encountered in evolution strategies (Schwefel, 1981). This kind of evolution maintains a population of μ formulae. Crossover and mutation are applied to this population of the size of μ to produce λ offspring. Selection then chooses the μ best performing formulae from the intermediate $(\mu + \lambda)$ to form the new generation. Selection is implemented as a k-tournament selection (Syswerda, 1991). Table 2 summarises the details for the runs. Before selection is applied to choose the μ formulae from the intermediate population $(\mu + \lambda)$, all clones (syntactically equal formulae) in the population are deleted. The elimination of clones can be efficiently implemented since the evaluation is performed in a directed acyclic graph (Handley, 1994; Keijzer, 1996), whereby clone investigation reduces to integer comparison.

3.2.2 Multi-objective approaches

In multi-objective settings, one considers a number of objectives. Solutions to multi-objective optimisation problems are expressed mathematically in terms of non-dominated or superior points. A solution is said to dominate another solution if it is not worse in any of the objectives and better in at least one. Obviously, in a multi-objective environment, a set of solutions must exist, a so-called 'front' of non-dominated solutions, or Pareto-front. In general, there is a number of issues that need to be addressed in multi-objective optimisation. For a review from an evolutionary perspective, the reader is referred to Srinivas and Deb (1994).

In the present multi-objective implementation a goodness-of-fit statistic — root mean squared error (RMSE) — and the goodness- of-dimension statistic were used.

In order to prevent premature convergence towards formulae with poor RMSE and perfect goodness-of-dimension, a simple, yet effective, mechanism was employed through the application of an auxiliary objective: a penalty on fitness similarity. Firstly, a check is performed on how many formulae have equal values on the other two objectives. The first of the encountered formulae is assigned a penalty of 0, the second a penalty of 1and so forth. Such fitness penalty is used as an auxiliary objective function that is used in the ranking process alongside the other objectives. This setup ensures that no two equivalent formulae can have the same rank on all three objectives. The k-tournament selection in the multi-objective approach is implemented as a tournament on dominance. Parents chosen in this way are then crossed (with or without guidance) and the resulting offspring is subsequently mutated (random only) to form one of the λ offspring.

During the selection process (from the intermediate population of $(\mu + \lambda)$ to μ), formulae are assigned a rank equal to the number of formulae that dominate it. The population is sorted on the basis of the rank in ascending order and subsequently reduced to the desired population size of μ. In this way, non-dominated solutions will remain in the population.

3.3 RESULTS

Several experiments have been performed, each increasing the amount of knowledge provided, either by adding information, or by removing unneeded (and potentially confusing) information.

For each of the experiments, the four versions of the genetic programming algorithm ran concurrently for the same amount of wall clock time. Because the number of individuals processed varies from setup to setup and from run to run, the usual graphs presenting the number of generations were not used. Instead, the results were binned into 25 buckets representing equal stretches of time, calculating aggregates for the values falling in the particular bucket. This provides a fairer picture of the actual effort involved — this method does not assume that every individual takes the same amount of time to be processed – a false assumption given the variable length nature of genetic programming. By using wall clock time instead of intricate calculations of the effort involved the

potential for making errors or false assumptions is effectively eliminated: if a method runs faster for whatever reason, it will be able to process more individuals.

3.3.1 Setup with randomly generated (dimensionless) constants only

In this setup, GP was allowed to use only randomly generated dimensionless constants. Therefore, a solution with both perfect goodness-of-fit and perfect goodness-of-dimension was unattainable. In this case, even if the perfect formula – the Bernoulli Law – were found, its goodness-of-dimension would equal 5. Note that an approach that only represents dimensionally correct expressions would get stuck in the very first generation with the only correct formulation possible: *z multiplied by a constant.*

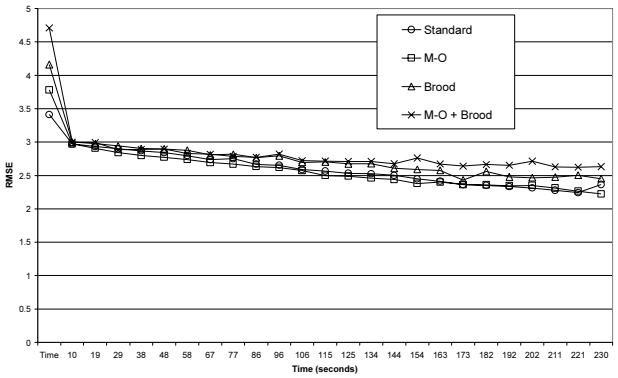

Figure 2. Results for an experiment with random constants only, averaged over 30 runs showing the RMSE of the best performing individuals for the 4 methods.

Figure 2 shows the RMSE evolution averaged over 30 runs. None of the 120 runs produced a perfect solution. Although the differences between the methods are very small, the multi-objective method (M-O) and the standard method produce the best results.

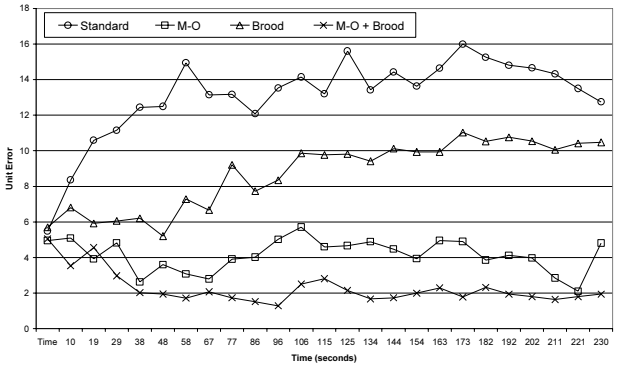

Figure 3. Average goodness-of-dimension for the best performing (RMSE) individuals in each run.

To illustrate the evolution of the goodness-of-dimension Figure 3 is presented. In both cases, brood-selection effectively reduces the magnitude of dimension errors made. In the present case where a perfect goodness-of-dimension is unattainable, this might hinder the search efficiency as seen in Figure 2.

Figure 4 illustrates a useful side effect of dimensionally-aware GP: promotion of parsimonious expressions. This figure illustrates that the best-performing individuals, averaged over 30 runs, are significantly different in size. Standard GP quickly fills up the available space, a phenomenon popularly referred to as bloating. The dimensionally aware methods do not grow as quickly, but rather utilise information contained in the units of measurements. Especially, the multi-objective methods result in smaller, dimensionally more correct expressions with a similar RMSE.

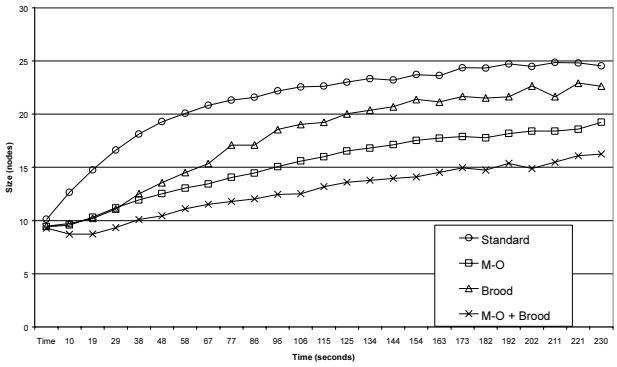

Figure 4 Results for an experiment with random constants only, averaged over 30 runs, showing the size of the best performing individual in population

3.3.2 Setup with randomly-generated and user-defined g and γ constants

In this series of experiments, the user-defined constants *g* and γ were added to the terminal set, all other parameters remaining the same.

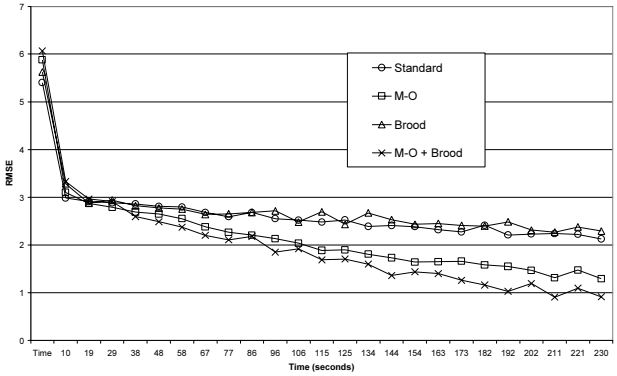

Figure 5. Results for an experiment with random constants together with user-defined constants g and γ, averaged over 30 runs showing the RMSE of the best performing individuals for the 4 methods.

The introduction of user-defined constants (g and γ) produced hardly any effect on the performance of the standard GP because these act merely as another two constant values, while the dimensionally-aware methods greatly benefited from the information so provided. The multi-objective methods now outperform the single-objective methods. Some optimal solutions were now found, see Table 3 for details. In this setup, the multi-objective methods produced expressions that were more accurate in both goodness-of-dimension and goodness-of-fit. In the single objective setup, the brood selection on goodness-of-dimension has a limited effect on the performance, while it has a much greater effect in the multi-objective setup.

3.3.3 Setup with user defined g and γ constants only

In this series of experiments, the random (dimensionless) constants were entirely removed from the terminal set. The problem now reduces to finding the proper formula using only z, v, p, g and γ. In such a rather constrained setup, perfect solutions were found on a regular basis. Table 3 provides more detailed information.

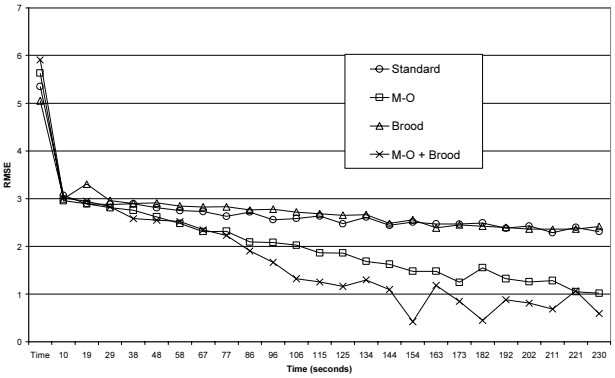

Figure 6. Results for an experiment with user-defined constants g and γ, averaged over 30 runs showing the RMSE of the best performing individuals for the 4 methods.

Table 3. Number of optimal solutions found in 30 runs (without $\mathbf{g} + \gamma$ no perfect solutions were found)

Method	**R + g + γ**	**g + γ**
Standard	0	0
Dimensionally-based brood selection	0	0
Multi Objective	1	13
Multi Objective with brood selection	2	20

It is slightly disappointing that only in the situation with the highly constrained experimental setup (the one without randomly created constants) the system is capable of finding the perfect solution on a regular basis. However, it is noteworthy that the combination of selection on goodness-of-dimension with variation using goodness-of-dimension produces the best results in the latter two experiments.

Strong points of the reported extensions are that dimensional awareness produces perfect solutions, and that dimensionally more correct formulae are produced consistently in all runs.

3.4 EXPERIMENTS ON NOISY DATA

In order to test the applicability of this method on noise-contaminated data, two additional data sets were created using 3% and 10% signal-to-noise ratios. These new, noisy data sets were characterised by non-smooth response surfaces and more pronounced local minima. The results obtained were qualitatively the same as those found for noise-free experiments: the techniques based on multi-objective fitness evaluation found perfect solutions. The evolution of the RMSE for the data with 3% noise and 10% noise are depicted in Figure 7 and Figure 8 respectively.

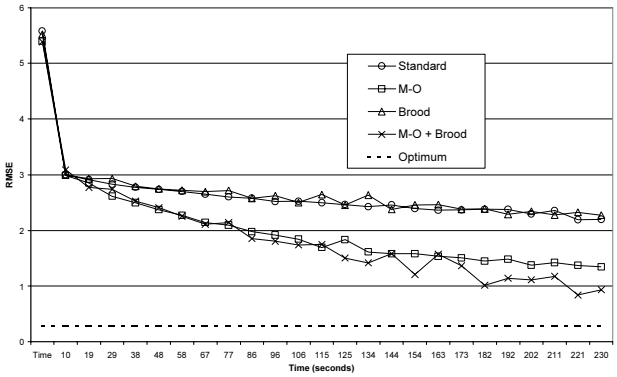

Figure 7. Results for an experiment on data with 3% noise contamination using R+ g + γ, averaged over 30 runs showing the RMSE of the best performing individuals for the 4 methods.

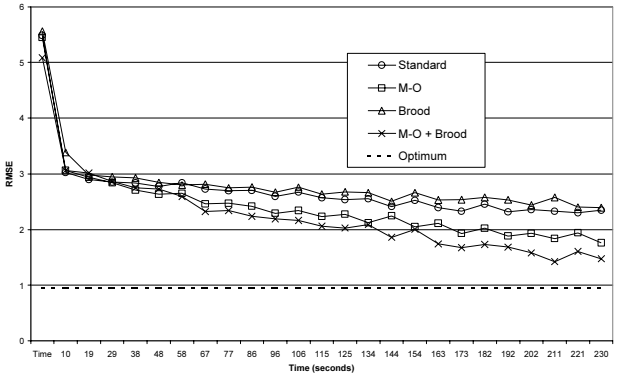

Figure 8. Results for an experiment on data with 10% noise contamination using R+ g + γ, averaged over 30 runs showing the RMSE of the best performing individuals for the 4 methods.

Such results are rather encouraging, since it is obvious that the information contained in the units of

measurement is helpful and can be effectively used in a search on noisy error surfaces. What is even more surprising is that these runs produced a perfect solution more often than the runs on non-contaminated data. These runs were performed using both type of constants and are thus comparable with the runs described in section 3.3.2. The runs involving noisy data had a success rate of one out of five or six, while the non-contaminated runs had a success rate of one or two out of thirty (see Table 3).

There is an added bonus in multi-objective approaches of this kind: it is possible to recognise an overfit without utilising test data. For example, when the perfect solution was discovered early in a run, continued evolutionary search produced formulae with even better goodness-of-fit. Noise in data allowed a better than optimal goodness-of-fit. Inspection of the final front of non-dominated solutions enabled an easy identification of the correct formulation as the one with the best goodness-of-dimension. The slight increase in goodness-of-fit of the overfitted formulae did not outweigh the much poorer dimensional fit.

4 DISCUSSION

The present contribution introduced an augmented version of GP that makes it more useful in the process of scientific discovery. Throughout science, the units of measurement of observed phenomena are used to classify, combine and manipulate experimental data. Exploitation of this information within GP makes this algorithm somewhat stronger as a search method. However, its applicability is broadened to encompass problems containing information on units of measurement.

The presented experiments constitute the first efforts in this direction. The results of the present work are encouraging since they consistently demonstrate the usefulness of dimensionally-aware GP. It has been shown that selection towards both goodness-of-fit and goodness-of-dimension produces the best results. It is also interesting that when a dimensionally correct formulation is attainable, brood-selection on goodness-of-dimension produces the best results when combined with the multi-objective approach, while it did not hinder the search in a single-objective approach.

Although this GP does not always improve goodness-of-fit, it invariably results in more parsimonious, closer-to-correct and easier-to-interpret formulations. Fundamentally, augmentation of GP with dimensional information adds a descriptive, semantic component to the algorithm. This is an addition to the functional semantics that defines the manipulation on numbers. While functional semantics grounds formulae in mathematics, the dimensional semantics grounds them in the physical domain.

Acknowledgements

This work was in part funded by the Danish Technical Research Council (STVF) under the Talent Project N° 9800463 entitled "Data to Knowledge — D2K". More information on the project can be obtained through http://projects.dhi.dk/d2k

References

Babovic, V., 1996, *Emergence, Evolution, Intelligence: Hydroinformatics*, Balkema Publishers, Rotterdam

Handley S., 1994., *On the use of a directed acyclic graph to represent a population of computer programs.* In Proceedings of the 1994 IEEE World Congress on Computational Intelligence , pp. 154-159, IEEE Press

Keijzer M., 1996., *Efficiently representing populations in genetic programming.* In Angeline, P.J. and Kinnear, K.E. Jr., (Eds), *Advances in Genetic Programming 2*, chapter 13, pp 259-278. MIT Press, Cambridge, MA

Koza J R., 1992., *Genetic Programming: On the Programming of Computers by Natural Selection,* MIT Press, Cambridge, MA

Schwefel, H.-P., 1981., *Numerical Optimisation of Computer Models*, Wiley, Chichester

Srinivas, N., and Deb, K., 1994, Multiobjective optimization using nondominated sorting in genetic algorithms, *Evolutionary Computation*, Vol 2, No 3, pp.221- 248

Syswerda, G.,, 1991. A study of reproduction in generational and steady state genetic algorithms in Rawlins, G., (Ed), *Foundations of Genetic Algorithms*, Morgan Kaufmann: San Mateo, CA.

Tackett, W. A.,1994., *Recombination, Selection, and the Genetic Construction of Computer Programs,*. Ph. D. thesis, University of Southern California