Helmut Hörner

Automated Generation of Physical Theories by a Genetic Programming Algorithm

Bachelor's Thesis

Vienna University of Technology

Institute for Theoretical Physics
Director: Ao.Univ.Prof. DI Dr. Gerhard Kahl

Supervisor: Ao.Univ.Prof. Dr. Karl Svozil

Vienna, October 2017

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all materials which have been quoted either literally or by content from the used sources.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

	10 October 2017
Mag. Helmut Hörner	Date

Abstract

This thesis describes how an Artificial Intelligence computer program in the domain of Genetic Programming can autonomously derive simple physical theories from (simulated) measurement data sets, where each data set describes the development of a specific physical system over time. After a brief introduction into Genetic Programming over k-bounded context-free languages, the structure and core functions of the specific software (developed by the author) and its algorithms are presented. Consequently, physical systems, each describing the kinetics of a point-like particle according to Newtonian Laws in environments of different complexity, are introduced. A simulated measurement data set with exact values, and also a data set with some Gaussian noise overlay is generated for each of the systems. Eventually it is shown to what extent the software (in its initial version, and in a version including some extensions programmed in course of this work) is able to derive the exact or an approximate physical model in the form of an analytic expression for each of these data sets on a powerful state-of-the-art desktop PC within a reasonable time frame.

Keywords: machine learning, genetic programming, induction, symbolic regression.

Contents

1 Introduction **8**
- 1.1 Motiviation . . . 8
- 1.2 Neural Networks vs. Genetic Programming . . . 9
- 1.3 Goal . . . 10
- 1.4 Method . . . 10
- 1.5 Outline . . . 11

2 A Brief History of Evolutionary Computation **11**

3 Genetic Programming over Context-Free Languages **12**
- 3.1 Derivation Trees, Genotypes and Phenotypes . . . 12
- 3.2 Generating a Random Population, and the Depth Limit Problem . . . 14
- 3.3 Fitness Calculation and Fitness Scaling . . . 15
  - 3.3.1 Raw Fitness . . . 15
  - 3.3.2 Standardized Fitness . . . 15
  - 3.3.3 Adjusted Fitness . . . 16
  - 3.3.4 Normalized Fitness . . . 16
- 3.4 Inclusion of Sub-Trees . . . 16
- 3.5 Selection . . . 17
- 3.6 Sampling . . . 18
- 3.7 Mating . . . 19
- 3.8 Cross-Over . . . 19
- 3.9 Mutation . . . 20
  - 3.9.1 Subtree-Mutation . . . 20
  - 3.9.2 Point Mutation . . . 20
- 3.10 Elitism . . . 21
- 3.11 Repetitive Application of all Genetic Operators . . . 21

4 Software Structure **21**
- 4.1 Genetic Programming Kernel . . . 21
- 4.2 Mathematical Expression Parser “NumEval” . . . 25
  - 4.2.1 Mathematical Operators . . . 25
  - 4.2.2 Mathematical Functions . . . 25
  - 4.2.3 Number Constants . . . 26
  - 4.2.4 Boolean Operators . . . 26
  - 4.2.5 Comparison Operators . . . 26
  - 4.2.6 Variables . . . 26
  - 4.2.7 Complexity . . . 27
- 4.3 Physical System Simulator “PhysModel” . . . 27
  - 4.3.1 User Functions . . . 27
  - 4.3.2 Internal Relations . . . 28

4.4 Top-Level Class InductiveLawDiscoverer 28
4.4.1 Raw Fitness Calculation 29
4.4.2 Using the Whole Software Package via InductiveLawDiscoverer . . 30

5 First Round of GP Simulations 32
5.1 Force-Free Point-Like Particle 34
5.1.1 Pre-Simulations 34
5.1.2 Reference Simulation Parameters 35
5.1.3 Simulation Result 36
5.2 Accelerated Point-Like Particle in Homogeneous Gravity Field 37
5.2.1 Pre-Simulations 37
5.2.2 Simulation Parameters 38
5.2.3 Simulation Result 39
5.3 Path of a Satellite Around a Central Gravitational Source in 1D Polar Coordinates 40
5.3.1 Simulation Parameters 40
5.3.2 Simulation Result 41

6 Conclusions After First Round and Consequently Introduced Software Improvements 42
6.1 Conclusions 42
6.2 Implemented Software Improvements 43
6.2.1 Multi-Threaded Raw Fitness Calculation 43
6.2.2 Multi-Threaded Parallel Gene Pools 43
6.2.3 Summary 43

7 Second Round of GP Simulations 46
7.1 Force-Free Point-Like Particle 46
7.1.1 Simulation Parameters 47
7.1.2 Simulation Result 47
7.2 Accelerated Point-Like Particle in Homogeneous Gravity Field 48
7.2.1 Simulation Parameters 48
7.2.2 Simulation Result 49
7.3 Path of a Satellite Around a Central Gravitational Source in 1D Polar Coordinates 49
7.3.1 Simulation Parameters 50
7.3.2 Simulation Result 51
7.4 Path of a Satellite Around a Central Gravitational Source in 2D Cartesian Coordinates 51
7.4.1 Simulation Parameters 52
7.4.2 Simulation Result 53

8 Conclusions After Second Round and Consequently Introduced Software Improvements **54**
8.1 Conclusions . . . 54
8.2 Implemented Software Improvements . . . 55
8.2.1 10-ary Gray Encoding . . . 55
8.2.2 Extended Grammar Definition Allowing For Non-Terminal Symbol Default Values . . . 56
8.2.3 Multiple Application of Cross-Over Operator and Mutation Operators . . . 57
8.2.4 Numerical "Endgame" Improvements . . . 57
8.2.5 Summary . . . 58

9 Third Round of GP Simulations **60**
9.1 Accelerated Point-Like Particle in Homogeneous Gravity Field . . . 60
9.1.1 Simulation Parameters . . . 60
9.1.2 Simulation Result . . . 60
9.2 Path of a Satellite Around a Central Gravitational Source in 2D Cartesian Coordinates . . . 62
9.2.1 Simulation Parameters . . . 62
9.2.2 Simulation Result . . . 62

10 Round Four: Simulating Noisy Measurement Values **64**
10.1 Force-Free Point-Like Particle . . . 64
10.1.1 Simulation Parameters . . . 64
10.1.2 Simulation Result . . . 64
10.2 Accelerated Point-Like Particle in Homogeneous Gravity Field . . . 68
10.2.1 Simulation Parameters . . . 68
10.2.2 Simulation Result . . . 68
10.3 Path of a Satellite Around a Central Gravitational Source in 1D Polar Coordinates . . . 71
10.3.1 Simulation Parameters . . . 71
10.3.2 Simulation Result . . . 71

11 Conclusions **74**

12 References **75**

13 Appendix **78**
13.1 List of Tables . . . 78
13.2 List of Figures . . . 79
13.3 Grammars . . . 80
13.3.1 Grammar A . . . 80
13.3.2 Grammar B . . . 81
13.3.3 Grammar C . . . 82

13.3.4 Grammar D 83
13.3.5 Grammar E 84
13.3.6 Grammar F 85
13.3.7 Grammar G 86
13.4 Simulation Protocols 87
13.4.1 Round 1, Simulation 1 87
13.4.2 Round 1, Simulation 2 88
13.4.3 Round 1, Simulation 3 99
13.4.4 Round 2, Simulation 1 100
13.4.5 Round 2, Simulation 2 100
13.4.6 Round 2, Simulation 3 103
13.4.7 Round 2, Simulation 4 105
13.4.8 Round 3, Simulation 1 106
13.4.9 Round 3, Simulation 2 117
13.4.10 Round 4, Simulation 1 122
13.4.11 Round 4, Simulation 2 132
13.4.12 Round 4, Simulation 3 149

1 Introduction

1.1 Motiviation

Creating mathematical models from quantitative observations has been at the heart of physics for many centuries.

Let us, just for a moment, consider an interesting historic example: In 1894, Max Planck was sitting in his study, brooding over data showing the spectral density of electromagnetic radiation emitted by a black body. He was, without success for many days and weeks, trying to find a formula that would explain the data. It must have been extremely frustrating. No progress at all. And then (as he later said) "in an act of despair" he decided to introduce the quantization of electromagnetic energy. And to his own surprise he so found the correct formula, now known as Plank's Law.

This story is inspiring. A brilliant, educated mind has successfully created a new physical theory. But as inspiring as this story may be, there is one core achievement: Plank's Law, which is basically an analytic mathematical expression that just "fits the data". In a very down-to-earth mind-set, one could say that Plank has "just" solved a curve-fitting, symbolic regression problem. Before Planck had found his new law, there was "unexplained" data; after he found it, there was the "correct" formula.

Of course, it must be questioned whether the "naked" formula is really the same thing as the physical theory. If we take *just* the formula, and do not *understand* where it comes from, then (as Friedrich Dürrennmatt put it in the mouth of Newton in his play "Die Physiker") we are rather just engineers, but no scientists[1]. To really gain knowledge, you either must know *how* Plank found his formula (he found it, because he assumed that electromagnetic energy comes in $E = hf$ quanta), or, if you don't know that, then you (just probably) may be able to reverse-engineer this insight from the formula.

We do not intend to solve this philosophical question. We settle on the fact that almost always a mathematical expression is at the core of a physical theory. We assume that the corresponding formula is a very important part of a physical theory, and that it would be interesting and useful to be able to derive such formulas from observational data by means of machine learning software.

[1] [Dürrennmatt, 1962, p. 19]: "Ich stelle nur auf Grund von Naturbeobachtungen eine Theorie ... auf. Diese Theorie schreibe ich in der Sprache der Mathematik nieder und erhalte mehrere Formeln. Dann kommen die Techniker. Sie kümmern sich nur mehr um die Formeln. ... Sie stellen Maschinen her, und brauchbar ist eine Maschine erst dann, wenn sie von der Erkenntnis unabhängig geworden ist, die zu ihrer Erfindung führte." - *"I am only devising a theory ... based on observations of nature. I write this theory in the language of mathematics and get several formulas. Then the engineers come. They only care about the formulas. ... They make machines, and a machine is useful only when it has become independent of the knowledge that led to its invention."*

1.2 Neural Networks vs. Genetic Programming

Two different machine learning methods were considered for this thesis: Neural Networks (nowadays better known under the buzzword "Deep Learning"), and Genetic Programming. Both technologies are known to be powerful and seem promising, but there is a major difference in applicability when considering the way we want our derived theories represented.

Theory Representation:

- A theory can be represented by a trained algorithm itself, if it directly has the ability to predict the future behavior of the physical system, i.e. by being able to predict the status of the physical system at any arbitrary point in time (see, for example [Svozil and Svozil, 2016]). We shall call this an *implicit physical model*.
- Alternatively, a trained algorithm could generate an explicit formula, an analytic expression, which allows the prediction of the behavior of the physical system for arbitrary points in time. This shall be called an *explicit physical model*.

As Genetic Programming algorithms are specifically designed for producing structured expressions based on grammars, they seem best suited for *explicit physical model* forming. On the other hand, neural networks are best known for solving regression and classification problems, rather than for creating a highly structured output (like creating a new formula from scratch).

If we are looking for an *explicit physical model*, then without further research it seems that the best we could do with a neural network is to assume a formula in advance with numerical parameters not yet defined (e.g. a polynomial expression), and then let the neural network find the best matching parameters. Alternatively, if we see the trained network as an *implicit physical model*, we could, in principle, translate this trained network into a formula, which would be again an *explicit physical model*. However, this formula would be ridiculously complex, and would definitely not provide any theoretical insight.

Therefore, and as we intend to demonstrate that computer software can actually *create* a new formula from scratch (without the need to make any assumptions about its structure in advance), we decided to use a Genetic Programming algorithm for this thesis.

1.3 Goal

Assume that we have a dataset of (exact or approximate) measurements describing the development of a physical system over time. As an example, this could be the (exact or approximate) positions x_1,x_2,...,x_n of a moving point-like particle at times t_1,t_2,...,t_n; where x_1,x_2,...,x_n may be scalars in a problem in on space dimension, or vectors in a multi-dimensional problem.

The **goal** of this thesis is to evaluate to what extent selected computer algorithms in the Genetic Programming domain are able to derive matching physical models from such data-sets on a state-of-the art desktop PC within reasonable time.

1.4 Method

As basis for our simulations we took the *Genetic Programming Kernel* (GPK) software package, developed in C++ by the author and described in [Hörner, 1996]. Since its original version, the author has expanded the GPK software so that it is now possible to not only calculate search space sizes of a given grammar, but also to ensure exact uniform random sampling with a given maximum derivation depth.

For this thesis, additional software development was done by the author:

- The GPK software package was modified as to compile into a 64bit executable under MS Visual Studio 2015.
- Multiple Grammars for creating BASIC-like mathematical expressions were developed.
- A new class `NumEval` was developed. An instance of this class represents a parser that is able to interpret BASIC-like mathematical expressions (the phenotypes).
- A class `PhysModel` has been implemented and tested. An instance of this class represents the physical model to be investigated by the AI. It receives as input a "secret" formula, which must be found by the AI, and generates a series of simulated measurements based on this (which can optionally be superimposed by Gaussian noise). The value range and the step size can be specified (separately for each parameter). In addition, the `PhysModel` class can compare phenotypes with the measured value series and then return a raw fitness value.
- A class `InductiveLawDiscoverer` has been implemented and tested. This class links the components described above, and forms the top level of the software in its first version.
- In the course of this thesis, as to improve performance, an additional software layer `LawDiscovererFarm` was developed, allowing runing of multiple GP simulations in parallel (multithreaded).

With this software package, we evaluated how efficiently GP simulations can deduce formulas for various kinetic problems. We did several "simulation rounds". After each round, the software was improved based on the experience gained. More details about the software structure can be found in Chapter 4.

The software was compiled with MS Visual Studio 2015 in "64 bit release" mode, and executed on an Intel i7 4.2 GHz 8-core desktop-PC with 64 GB RAM under Windows 10 (Build 15063).

1.5 Outline

Chapter 2 is a brief history of evolutionary computation. Chapter 3 is where we explain how tree-based GP algorithms work in general. To keep this thesis compact, and to not distract the reader, we focus specifically on methods and genetic operators that were actually used in our simulations. Chapter 4 gives details about the software structure with regard to the version of the software used for the first simulation round. In the succeeding chapters we present the result of each simulation round, always followed by a critical review and explanations on how the software (or the grammar) was modified as to improve the simulation results. The thesis eventually concludes with a discussion of the results, and an appendix containing references, used grammars, and protocols.

2 A Brief History of Evolutionary Computation

Artificial Intelligence (AI) offers a wide variety of approaches for machine learning. Genetic Programming (GP) is one of these numerous approaches, and it belongs to AI sub-field of "Evolutionary Computation" (EC). EC includes all computer algorithms that intend to solve an optimization problem by imitating biological evolution. Typically, in such algorithms an initial set of (often randomly generated) "individuals" - each representing a solution to the given problem - have to compete against each other, and are subjected to selection and evolution processes. The idea is that, like in nature, over time better and better individuals will emerge, until eventually an optimal, or at least close-to-optimal solution is found.

The roots of EC go back to the 1960s: In the U.S., Lawrence J. Fogel introduced EC as a method for time-series predictions [Fogel et al, 1966]. In 1975, John H. Holland published his important book on "Genetic Algorithms" ("GA", [Holland, 1975]), thereby defining the term "GA". Pioneering work in the area was also done in the 1970s in Germany by Ingo Rechenberg [Rechenberg, 1973], who named his method "Evolution Strategy". Besides others he was able to optimize aerodynamic wing designs by his version of EC.

Although these early approaches where quite successful, they all where challenged by the problem of how to encode the potential solutions into the simulated "individuals". You have to find some meaningful encoding, so that after applying all evolutionary operators

(like mutation, or mating), you still get resulting individuals that represent meaningful ("syntactically correct") solutions.

At that time, GA "individuals" typically consisted of mere bit-strings of a fixed length. Depending on the problem at hand, you could, for example, interpret groups of 8 bits as numbers, where each number encodes a numerical parameter of the system. Or, if your problem is to find an optimal path on a rectangular grid, you could take groups of two bits, encoding step-by-step the directions "forward", "backwards", "left" and "right".

However, what if the solution you are looking for is (like in our case) a mathematical expression of unknown complexity, or even a whole computer program? Seemingly, the above-mentioned simple "GA"-type of encoding has reached its limit here.

A solution to this challenge was first presented in 1985 by Nichael Lynn Cramer [Cramer, 1985]. A couple of years later Cramer's approach was vastly expanded by John R. Koza, who also coined the expression "Genetic Programming" (GP) for this new method. Since then, GP has been applied very successfully to numerous complex optimization and search problems [Koza, 1992], and some solutions found by GP algorithms have even been patented.

3 Genetic Programming over Context-Free Languages

This chapter explains how tree-based GP algorithms work in general, but always keeps a specific focus on the actual software implementation and the chosen parameters used for the simulations in the course of this thesis.

3.1 Derivation Trees, Genotypes and Phenotypes

With GP programs, "individuals" usually are no longer mere strings of bits or symbols, but usually complete derivation trees of a specific grammar. Let us, for example, take the following simple grammar, given in its Backus-Naur representation:

```
S := <expression>;
<expression> := <variable> | <expression> <operator> <expression>;
<operator> := "+" | "-" | "/" | "*";
<variable> := "x" | "y" | "z";
```

Figure 1: A simple grammar.

Note that, deviating from the common Backus-Naur representation as suggested in [Knuth, 1964], we just write `:=` instead of `::=` (for sake of simplicity). The Symbol `S` indicates where to start the expansion. All symbols in pointed brackets shall be called *non-terminal symbols* (because they need to be further substituted), and all symbols in

quotation marks shall be called *terminal symbols* (because these are the symbols where a leaf of the derivation tree ends).

Starting with symbol `S`, we can easily create a complete derivation tree from this example grammar; as shown in Figure 2:

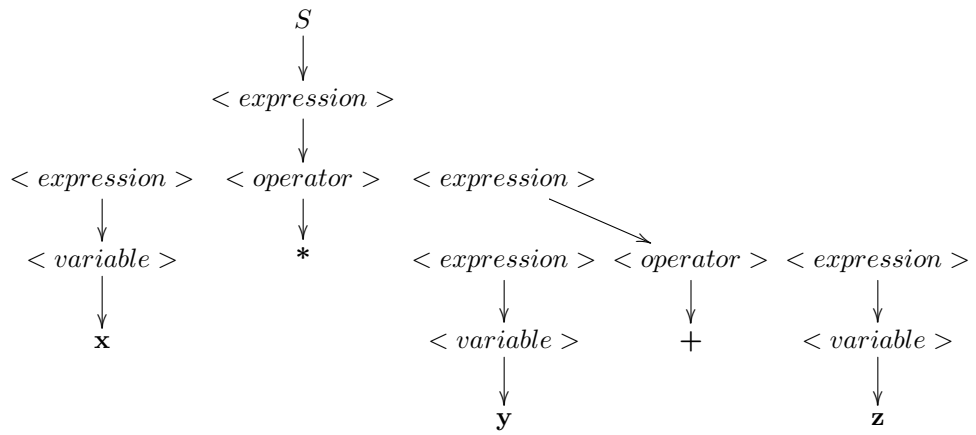

Figure 2: A GP "individual", represented by a simple derivation tree.

Interestingly, there are two ways to interpret the "individual" in Figure 2. If you concentrate on the derivation tree and follow its node expansions , then the sub-expression `y+z` merges up-tree into one node before the operator $*$ becomes active, and therefore `y+z` is to be calculated first. Hence, in this interpretation, the derivation tree represents the expression `x*(y+z)`. For maximum speed and efficiency, GP software implementations very often integrate the parsing process into the derivation tree, and then this is the result you get.

On the other hand, it is good software design practice to strictly separate software layers. From this perspective, mixing the derivation tree and the parsing process is "messy". To ensure maximum modularity, it makes a lot of sense to clearly distinguish between the derivation tree (the so-called *genotype*), and the resulting expression (the so-called *phenotype*). In that case, the *phenotype* `x*y+z` can be interpreted completely independent from its *genotype* by a separate parser, and is - according to the common rule that multiplication has precedence before addition - to be interpreted as `(x*y)+z`. This is the approach taken by the software implementation discussed in this thesis.

It should be mentioned that there is an alternative approach to overcoming the encoding problem mentioned at the end of Chapter 2. It is called *Linear Genetic Programming* (LGP) and works without tree-representations under the condition that specific programming languages are used. In this thesis we will not further investigate LGP.

3.2 Generating a Random Population, and the Depth Limit Problem

The first step in a GP algorithm is to generate a population of random individuals. But even if you consider the very simple grammar as shown in Figure 1, it is obvious that because of the recursive nature of the grammar, an infinite amount of (ever larger) individuals can be derived from it.

Computing the exact number of individuals that can be generated in a certain number of steps from a given (maybe complex) grammar, and consequently computing the total number of individuals that can be generated for a maximum derivation depth limit, and doing so within a reasonable time frame, is in general a non-trivial task.

Luckily, although not entirely trivial, it is a task that has been already solved. In [Geyer-Schulz, 1994], an APL algorithm for calculating the number of individuals that can be derived in n steps for a given grammar was presented for the first time. Unfortunately, the proposed implementation was too slow for most practical purposes. In [Hörner, 1996, p. 22-25], a much faster algorithm and its C++ implementation was presented, and is consequently also used in this work.

The following table is calculated by this algorithm, and shows how many individuals can be created in *exactly* n derivation steps, and also how many individuals can be created *in total* with a given *maximum* of n derivation steps from the sample grammar in Figure 1 (substitution of starting symbol S not included):

No of steps n	**No. of individuals that can be created in exactly n steps**	**Total no. of individuals that can be created in maximum n steps**
2	3	3
6	36	39
10	864	903
14	25 920	26 823
18	870 912	897 735
22	31 352 832	32 250 567
26	1 182 449 664	1 214 700 231
30	46 115 536 896	47 330 237 127
34	1 844 621 475 840	1 891 951 712 967
38	75 260 556 214 272	77 152 507 927 239
42	3 119 892 148 518 912	3 197 044 656 446 151
46	131 035 470 237 794 304	134 232 514 894 240 455
50	5 563 967 659 327 881 216	5 698 200 174 222 121 671

Table 1: Number of individuals that can be created from the sample grammar in Figure 1 in up to 50 derivation steps.

As the number of individuals that can be created is unlimited, one has to set an allowed maximum derivation depth. If you do so, the rightmost column of Table 1 gives you the sample size of all individuals that can possibly be created in n steps. That can be seen as the *search space size*.

If there is no hint which individuals are to be preferred, a natural strategy is to create an initial population where each individual of the search space has equal probability to be randomly created. In [Hörner, 1996, p 25-29], a time-efficient algorithm for achieving this task is presented, and it has been implemented by the author in the GPK software package used for this thesis.

However, if you look at the numbers in Table 1, you see that there are always overwhelmingly more individuals that can be created for your chosen maximum derivation depth n_{max} than for all smaller derivation depths $n < n_{max}$ together. This is typical for all relevant grammars, and implies that, by choosing a maximum derivation depth n_{max}, you practically define the size of almost all individuals in the initial population. This is a problem, as we usually don't know the size of the solution we are looking for. We will refer to how we approached this problem in Chapter 5.

3.3 Fitness Calculation and Fitness Scaling

3.3.1 Raw Fitness

The next step in any GP algorithm is to assign a *raw fitness value* f_i^{raw} to each individual i. This *raw fitness value* must give an indication on how "fit" the individual is with respect to the given problem. In our case, the mean square deviation between the values calculated by the individual's *phenotype* and the measurement value data set is a good candidate for calculating f_i^{raw}. Actually, the mean square deviation is the basis for f_i^{raw} in our simulations, but also some "extras" have been added to calculating f_i^{raw}. We will elaborate the details in Chapter 4. Depending on the problem, f_i^{raw} is either to be maximized, or minimized. In our case, of course, it must be minimized.

3.3.2 Standardized Fitness

As the GPK software is for general use, and - depending on the task on hand - f_i^{std} must be either maximized or minimized, a so-called *standardized fitness value* f_i^{std} is calculated for each individual i in the next step. As proposed by [Geyer-Schulz, 1994, p. 218], it is calculated differently for maximization tasks and minimization tasks, as to ensure that we get a fitness value where smaller values *always* indicate better individuals. For minimization tasks the standardized fitness is calculated as follows:

$$f_i^{std} = f_i^{raw} - \min_{i=1...n} f_i^{raw} \tag{1}$$

where $\min_{i=1...n} f_i^{raw}$ represents the smallest raw fitness value in the whole population at the time.

3.3.3 Adjusted Fitness

Next, the so-called *adjusted fitness value* f_i^{adj} is calculated for each individual i from f_i^{std}. It always has a value between 0 and 1, where larger values indicate fitter individuals. Any linear or non-linear function assigning larger values to fitter individuals which ensures that f_i^{adj} always lies between 0 and 1 is suitable. If the selection of individuals later is done proportional to the (normalized) fitness values (which are calculated from f_i^{adj}), this function influences the selection probability. It can be used, for example, to eliminate non-linearity of f_i^{raw}. In our implementation we have chosen the following function, as suggested by [Koza, 1992, p. 97]:

$$f_i^{adj} = \frac{1}{1 + f_i^{std}} \tag{2}$$

3.3.4 Normalized Fitness

Eventually, the *normalized fitness value* f_i^{norm} is calculated for each individual i according to [Geyer-Schulz, 1994, p. 218], so that the sum over all normalized fitness values equals 1:

$$f_i^{norm} = \frac{f_i^{adj}}{\sum_{i=1}^{n} f_i^{adj}} \tag{3}$$

Again, the larger the value, the fitter the individual.

3.4 Inclusion of Sub-Trees

Consider the example derivation tree from Chapter 3.1, which produced the phenotype `x*(y+z)`. Maybe this phenotype has a rather bad fitness value. However, if we also consider all sub-trees starting with root-node `<expression>`, then we can additionally extract the following syntactically correct phenotypes from the derivation tree:

- `x`
- `y+z`
- `y`
- `z`

It is easily possible that some of these expressions have a better fitness than the complete phenotype `x*(y+z)`. Hence, it seems reasonable to also apply fitness values to some or all of these sub-expressions, and to create a "virtual" population that includes these additional phenotypes.

Because of this reasoning, the GPK software package offers the possibility to activate the inclusion of sub-trees with a defined minimum derivation size. However, multiple simulations run by the author indicated that permanent activation of this feature is not entirely beneficial and can lead to premature convergence. Therefore, in the actual simulations we only activated this feature once every 10 generational steps, which seemed to have a positive effect on the overall performance.

3.5 Selection

After calculating the fitness values for each individual in the population, a *target sampling rate* tsr_i is to be calculated for each individual i. It gives the expected value for how often this individual i will be (on average) selected in the following random sampling process.

At first glance, calculating tsr_i somehow proportional to f_i^{norm} seems reasonable, and it is indeed a method suggested by various authors (e.g. [Goldberg, 1989, p. 10ff], or [Schöneburg et al., 1994, p. 197ff]). However, this method has a serious drawback: When the population is very diverse (e.g. at the beginning), the fittest individuals get a much higher chance of being selected than all others, and consequently tend to take over the whole population very soon. On the other hand, if (after a while) the population gets more and more homogeneous, the selection pressure drops, and the evolutionary process degrades to a mere random process.

To overcome these problems, [Grefenstette et al., 1989] suggested the method of *Linear Rank Selection*, and since then this method has been widely adopted in the GP community (see: [Poli et al., 2008, p. 32-34]). With this method, the *target sampling rate* tsr_i is calculated as follows:

$$tsr_i = tsr_{min} + (tsr_{max} - tsr_{min})\frac{rank_i - 1}{n - 1} \tag{4}$$

where n is the total number of individuals in the population, $rank_i$ is the rank of the i-th individual if ordered by f_i^{norm} in ascending order (starting with 1), and tsr_{min} denotes the (chosen) target sampling rate for the unfittest individual. tsr_{max} must fulfill the following condition:

$$tsr_{max} = 2 - tsr_{min} \tag{5}$$

3.6 Sampling

In the next step, a sample of individuals is to be drawn randomly in n drawings, where n denotes the size of the new population. In the simplest case of *Stoachastic Sampling with Replacement* the probability for the i-th inividual to be drawn is calculated by

$$p_i = \frac{tsr_i}{\sum_{j=0}^{m} tsr_j} \tag{6}$$

where m denotes the number of individuals in the *current* population.

Of course, what can easily happen here, is that some individuals with a tsr around 1 have "bad luck" and are not sampled at all. To prevent this from happening, and to ensure maximal diversity, [Baker, 1987] suggested *Stochastic Universal Sampling*. That is the sampling method we have used in all our software simulations. The basic idea of *Stochastic Universal Sampling* is to sample all individuals in *one* drawing.

Let's assume, for example, that we have a current population of size $m = 3$, and we want to sample 5 individuals ($n = 5$). For the Stochastic Universal Sampling process, we can visualize a roulette wheel with 3 compartments, each sized according to the respective target sampling rate (see Figure 3).

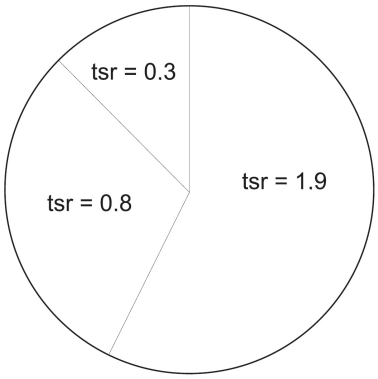

Figure 3: "Roulette wheel" with 3 compartments, each sized according to the respective target sampling rate, from: [Hörner, 1996, p. 34].

For the actual sampling process, we put another wheel with n spokes on top of the first one, and we spin it until it comes to a stop at a random position (see Figure 4).

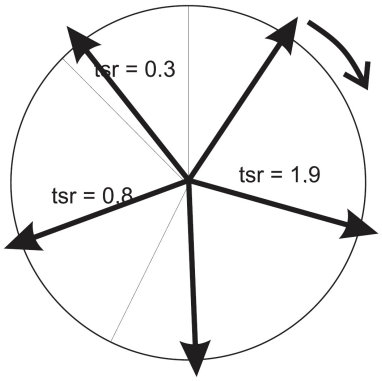

Figure 4: Stochastic Universal Sampling of 5 individuals, from: [Hörner, 1996, p. 34].

In our example, as you can see from Figure 4, the sample consists of three copies of the individual with $tsr = 1.9$, and one copy of each for the other two individuals.

3.7 Mating

The next step in a GP algorithm is to find a mating partner for each individual in the previously generated sampling list. The mating method we have chosen for all of our simulations is *Random Permutation Mating*, as suggested by [Geyer-Schulz, 1994]. With this method, a random permutation of the previously generated sampling list is generated and assigned to the original list. Then, the i-th entry of the permutated list is the assigned mating partner of the i-th entry in the original list.

3.8 Cross-Over

For each pair in the mating-list there is a certain probability (chosen as a parameter for the simulation), that the following cross-over operator is applied (thereby generating a new population consisting of individuals forming - after applying some more genetic operators - the next generation):

- Count all tree-nodes representing *non-terminal symbols* (see Chapter 3.1) in the derivation-tree of the first mating partner.
- Select one of these nodes at random and remember the symbol.
- Find all tree-nodes of the second mating-partner having exactly the same symbol, and having a sub-tree small enough to potentially replace the sub-tree of the selected node in the first partner without exceeding the maximum derivation depth. If no such node exists, repeat with step 1 until a maximum of 8 attempts have been reached.

- Select one of the identified suitable nodes of the second mating-partner by random.
- Replace the sub-tree starting with the selected node of the first mating partner with the sub-tree starting with the selected node of the second mating partner.

If a pair is not chosen for cross-over, then the first mating partner is copied into the new population without any modification at that step.

3.9 Mutation

After mating and cross-over, mutation operators are usually applied in GP software. There is an ongoing discussion to what extent mutation operators actually improve the performance of GP. [Luke and Spector, 1997] suggested that the performance gain due to mutation very much depends on the problem and the detail of the GP system.

Multiple mutation operators are described in literature (see [Poli et al., 2008, p.42-44], for example: *Subtree Mutation, Size-Fair Subtree Mutation, Point Mutation, Hoist Mutation, Shrink Mutation, Permutation Mutation, Random Constant Mutation, or Systematic Constant Mutation*, to name just a few.

In our application we use *Subtree Mutation*, and *Point Mutation*, which are described in this chapter.

3.9.1 Subtree-Mutation

For each individual in the new population, which was created by mating and cross-over, there is a certain probability (chosen as a parameter for the simulation) that the subtree-mutation operator is applied.

The subtree-mutation operator simply replaces a randomly selected sub-tree (that's any tree starting with a *non-terminal symbol*) with a randomly generated sub-tree, while ensuring that the maximum derivation depth is not exceeded.

3.9.2 Point Mutation

After the tree-mutation operator has been applied, there is a certain probability (chosen as a parameter for the simulation) that the point mutation operator is applied.

From all *terminal symbols* of an individual's derivation-tree the point mutation operator chooses one at random, and replaces it with a randomly selected different terminal symbol allowed by the grammar. One of the advantages of the point mutation operator is that it helps evolving and fine-tuning number constants in the individual's phenotype.

3.10 Elitism

The elitism-operator, if activated, is applied after all other operators. It ensures that there is always at least one copy of the previous generation's best individual in the next generation of individuals.

3.11 Repetitive Application of all Genetic Operators

If all operators as described in Chapters 3.3 to 3.10 are applied to the initial random population, the first new (and hopefully fitter) generation has been created. After that, repetitive application of these operators lead to populations with ever fitter individuals, hopefully converging into an approximate or even exact solution to the given problem.

As described, there are a lot of parameters to be chosen, e.g. population size, maximum derivation depth, or various probabilities for cross-over and mutation operators, to name just a few. Choosing these parameters is an meta-optimization task in itself, and finding well-performing parameters often depends on the experience of the person running the simulations. It is also not uncommon to vary some of these parameters during the simulation process.

4 Software Structure

This chapter explains design and structure of the used software package. We hereby refer to the initial version of the software that was used in the first simulation round. Any improvements that have been implemented in course of later simulation rounds are explained in Chapters 6.2 and 8.2.

The UML class diagram in Figure 5 shows the most relevant classes of both the Genetic Programming Kernel (that's everything within the grey box), as well as of the software that has been implemented specifically for this thesis (that's everything displayed above the grey box).

4.1 Genetic Programming Kernel

[Hörner, 1996] gives an extended description of the Genetic Programming Kernel (GPK). As this thesis focuses on an GP application, using the GPK just as an tool, and as the general foundations of GP have been outlined in Chapter 3, a brief overview about what's going on "under the GPK's hood" should be sufficient here.

- Class `Farm` ist the GPK's top class. It stores the population (or, optionally, several populations, representing multiple generations), the grammar, and all simulation parameters. Further, it gives top-level access to functions for configuring the simulation parameters, like `.SetFitnessScaling(...)`, `.SetTreeMutProb(...)`, etc., and the `.NextStep()` function, triggering the next generational step.

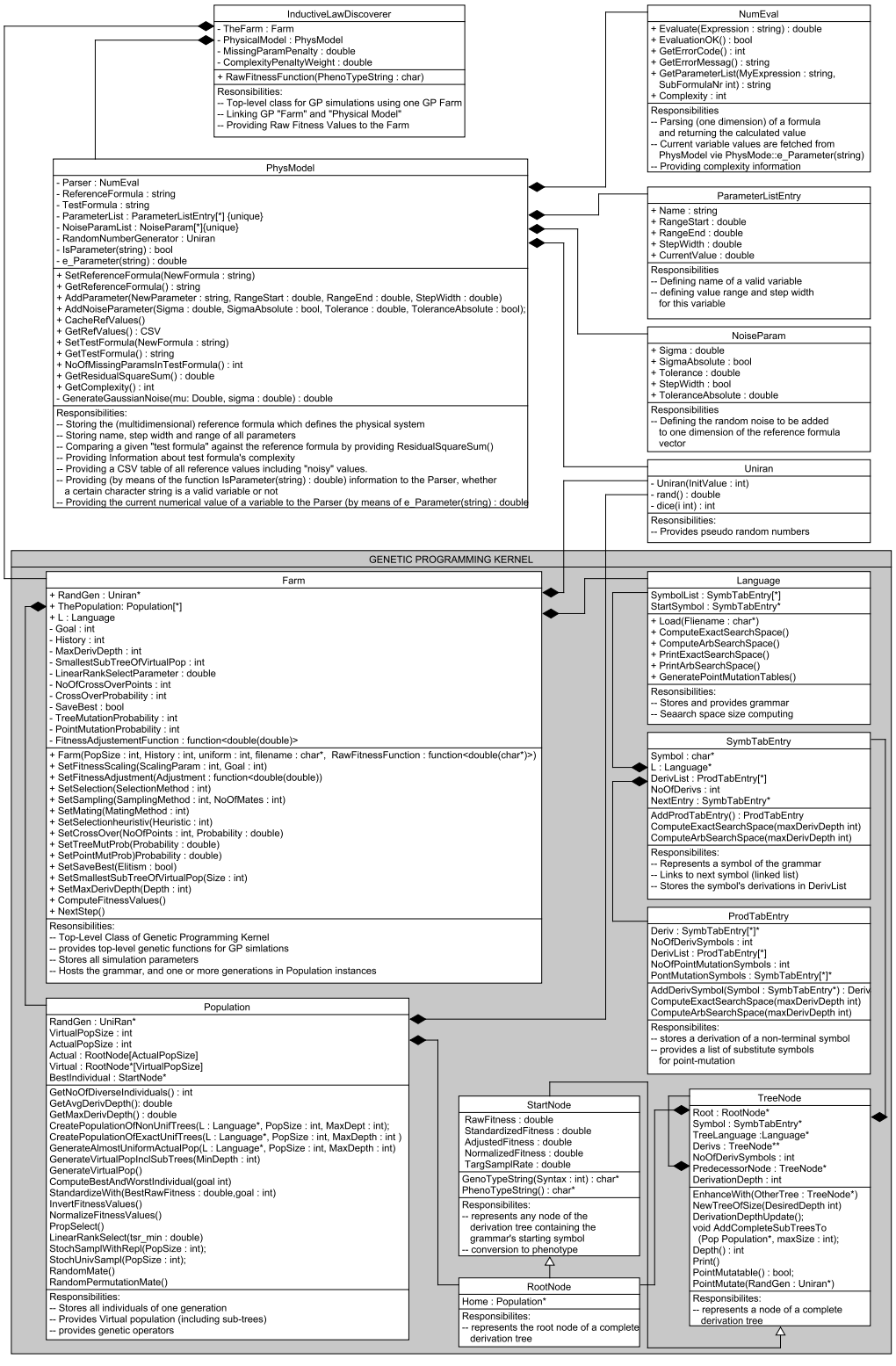

Figure 5: UML diagram of the initital software version.

- Class `Language`, of which an instance is stored in every instance of class `Farm`, is able to load a grammar from a grammar definition file, and provides access to all symbols and its derivations (with helper classes `SymbTabEntry`, which instances represent terminal- or non-terminal-symbols, and class `ProdTabEntry`, which instances represent derivations of non-terminal symbols). Further, class `Language` and its helper classes provide functions for search space size computing, as described in [Hörner, 1996].

- Instances of class `Population` represent a population of individuals in a specific generation. Further (if activated), this class generates a "virtual population", including also all sub-trees down to a specified minimal derivation depth (see Chapter 3.4).

- Any instance of class `RootNode`, derived from class `StartNode`, represents a root node of a complete derivation tree.

- Any instance of class `StartNode`, derived from class `TreeNode`, represents a node of a derivation tree containing the grammar's top-most non-terminal symbol (after the starting symbol `S`). This class, for example, provides the public function for reading out a phenotype string.

- Eventually, any instance of class `TreeNode` represents any other node of a derivation tree. Of all the node-classes, this class contains most of the helper functions needed by the genetic operators, like "growing" a random tree, point mutation, and much more.

The pseudo number generator class `Uniran` is used by both the GPK itself, as well as by the newly developed class `PhysModel`. It is based on an PASCAL implementation described in [Law and Kelton, 2000, p. 428], and provides up to 255 independent pseudo random number streams.

The following source code demonstrates how to use the "naked" GPK on top level, and how to start a GP simulation in a few lines of code.

```
1  #include "ga.h"
2  #define MAXDERIVDEPTH 70
3  #define SMALLESTSUBTREE 20
4  #define POPSIZE 10000
5  #define HISTORY 1
6  #define GPKRANDSEED -1
7  #define LANGUAGEDEF "c://temp/MyGrammar.def"
8  #define TARGETFITNESS 0.01
9
10 int main()
11
12 {
13    Farm MainFarm(POPSIZE, HISTORY, EXACTUNIF, LANGUAGEDEF, MAXDERIVDEPTH,
14                  SMALLESTSUBTREE, GPKRANDSEED);
15    MainFarm.SetFitnessScaling(0, MINIMIZE);
16    MainFarm.SetFitnessAdjustment(One_DividedBy_OnePlusX);
17    MainFarm.SetSelection(LINEAR_RANK, 0.5);
18    MainFarm.SetSampling(STOCHASTIC_UNIVERSAL, POPSIZE);
19    MainFarm.SetMating(RANDOM_PERMUTATION);
20    MainFarm.SetSelectionHeuristic(PropSelectionHeuristic, SEARCHSPACESIZE);
21    MainFarm.SetCrossOver(1, 0.2);
22    MainFarm.SetTreeMutProb(0.04);
23    MainFarm.SetPointMutProb(0.03);
24    MainFarm.SetSaveBest(YES);
25
26    int i;
27    i = 0;
28    while(true)
29    {
30       i++;
31       cout << endl << i << ": Fitness ";
32       cout << MainFarm.Current->BestIndividual->RawFitness << ": ";
33       MainFarm.Current->BestIndividual->Print();
34
35       if(MainFarm.Current->BestIndividual->RawFitness<=TARGETFITNESS)
36          break;
37
38       MainFarm.NextStep()
39    }
40 }
```

Figure 6: Sample source code, demonstrating how to use the "naked" GPK.

The above listing is easy to understand. In line 13, a new `Farm` is instanced, with `POPSIZE 10000` individuals. `HISTORY 1` means that genetic operators do not include previous generations, `EXACTUNIF` forces the GPK to create a uniformly distributed population, and `LANGDEF` specifies from where to load the grammar definition file.

`MAXDERIVDEPTH 70` limits the maximum derivation depth to 70 steps, `SMALLESTSUBTREE 20` means that all sub-trees with minimum derivation depth of 20 steps should be included into the selection process. The last parameter `GPKRANDSEED -1` is just the seeding value for the random number generator.

In lines 15-24, all other simulation parameters are defined. For example, line 15 defines that fitness scaling should only consider the current population (first parameter = `0`), and that the goal is to `MINIMIZE` the raw fitness (second parameter). Or, as another example, line 21 defines that the cross-over parameter should do cross-over in one point (first parameter), and be applied with a probability of `0.2` (second parameter).

Eventually, the main loop from line 28-39 prints out the currently best individual (and its fitness), and triggers the next generation (line 38). When the best individual is better than the desired target fitness, the loop is abandoned (lines 35 and 36).

However, there is one important topic that we have circumvented in this explanation so far: The above code will *only* work if we have added some code to evaluate the phenotypes, and to calculate the raw fitness value for each individual. This will be the subject of the next sub-chapters.

4.2 Mathematical Expression Parser "NumEval"

Any instance of `NumEval` is immediately able to evaluate BASIC-lime mathematical expressions. The expression must be passed by calling the function `.Evaluate(string)`, which returns the numerical value. If an error has occurred, `.EvaluationOk()` returns `false`, and `.GetErrorCode()` and `.GetErrorMessage()` provide further information about the problem. The parser works case-sensitive.

4.2.1 Mathematical Operators

The following common mathematical operators, considering the usual operator precedents, are included:

```
+, -, *, /, ^
```

The minus-sign can also act as an unary operator, reversing the sign of the expression on the right side.

4.2.2 Mathematical Functions

Further, the following common mathematical functions, all accepting exactly one parameter, can be interpreted (where `<num>` stand for any syntactically correct numerical expression):

```
Sin(<num>), Cos(<num>), Tan(<num>), Abs(<num>), Exp(<num>), Log(<num>),
ArcSin(<num>), ArcCos(<num>), ArcTan(<num>), Sinh(<num>), Cosh(<num>),
Tanh(<num>), ArcSinh(<num>), ArcCosh(<num>), ArcTanh(<num>), Sqrt(<num>)
```

Also, a Min and a Max function has been included, both of which expect two numerical parameters:

```
Min(<num>, <num>), Max(<num>, <num>)
```

The numerical constant π is included as pseudo-function:

```
Pi()
```

4.2.3 Number Constants

Any string of digits with maximum one point in-between will be interpreted as number. It is also possible to use numbers in scientific notation: Any number as defined before followed by an `E`, optionally followed by `+` or `-`, and then followed by one or two further digits will be interpreted as a number in scientific notation.

4.2.4 Boolean Operators

The `NumEval` class is also able to handle boolean algebra. The numerical value `0` is interpreted as `false`, and any other numerical value as `true`. These are the implemented boolean operators, acting on the boolean value on the left and the right side, and returning a boolean value:

```
And, Or, Xor
```

The "Not" operator is implemented as function, accepting exactly one (boolean) value:

```
Not(<bool>)
```

4.2.5 Comparison Operators

The following comparison operators, comparing the numerical value on the left and on the right side, and returning a boolean value, are also implemented:

```
<, >, >=, <=, <>, =
```

This allows for the implementation of an `Iif`-function, which expects three parameters: One boolean, and two numerical:

```
Iif(<bool>, <num>, <num>)
```

4.2.6 Variables

As the GP algorithm should eventually produce mathematical functions, which necessarily depend on one or more variables, the `NumEval` class must be able to also interpret these variables (like, for example `t`, for time). Therefore, it can be initialized with two function pointers of type `bool IsParameter(string)` and `double e_Parameter(string)`. These functions must be provided externally (in our case they are provided from class `PhysModel`). Whenever the parsing algorithm comes across a potential variable, it first calls `IsParameter(string)` to find out, whether this is a valid variable at all. If yes, `e_Parameter(string)` is called to fetch the current value of this variable.

4.2.7 Complexity

After having evaluated a formula, the public integer variable `.Complexity` provides information about the function's complexity. This is how this value is calculated:

It comes naturally that the parser works recursively. Every time the internal parsing function `.e_prs(string)` is called recursively, the value of `.Complexity` is increased by one. So, for example, while evaluating the expression

```
Sin(Sqrt(t+2))
```

the parsing function is called five times:

- `.e_prs("Sin(Sqrt(t+2))")`
- `.e_prs("Sqrt(t+2)")`
- `.e_prs("t+2")`
- `.e_prs("t")`
- `.e_prs("2")`

Hence, the expression `Sin(Sqrt(t+2))` has a `.Complexity` value of 5.

4.3 Physical System Simulator "PhysModel"

4.3.1 User Functions

An instance of class `PhysModel` represents the to-be-discovered physical system. First of all, it stores the "secret" formula, which represents the system. This formula can be set with `.SetReferenceFormula(string)`. The formula can have one or more variables, and it can represent both a scalar or an vector. For the latter, simply multiple expressions are concatenated with an "`:`". For example, `"Cos(phi):Sin(phi)"` would represent an unit circle in polar coordinates.

Then, all valid parameters (variables), plus their value ranges, plus the step-width for the to-be-generated value list must be defined. This is done by means the function `.AddParameter(...)`. In our unit circle example, we could set the parameter φ to run from 0 to 2π in steps of 0.1 by calling `.AddParameter("phi", 0, 2*3.141, 0.1)`.

Optionally, it can be defined if (and if yes, how much) Gaussian random noise (simulating measurement errors) should be superimposed on the values. This is done by calling the function `.AddNoiseParam(..)`, which accepts 4 parameters. The first parameter `double Sigma` is the sigma value for the Gaussian distribution. The second value `bool SigmaAbsolute` defines, how Gaussian distributed random numbers

GaussRnd($Sigma$) are superimposed on the simulated measurement values v. If `SigmaAbsolute == true`, then the random number is added as follows:

$$v_{noisy} = v + \text{GaussRnd}(Sigma) \tag{7}$$

If `SigmaAbsolute == false`, then the "noisy" value v_{noisy} is calculated this way:

$$v_{noisy} = v \cdot (1 + \text{GaussRnd}(Sigma)) \tag{8}$$

The last two parameters are `double Tolerance` and `bool .ToleranceAbsolute`. If during the calculation of the mean square deviation over all reference values the difference between a (noisy) reference value and the to-be-tested function is smaller than the `Tolerance` value, then for this point a deviation of zero is assumed. The `ToleranceAbsolute` parameter defines, whether the `Tolerance` value is meant as an absolute value, or as an relative tolerance.

After having defined the system, the reference value list is generated by calling `.CacheRefValues()`. The function `.GetRefValues()` can be used to export the whole reference value table (including all noisy values) in CSV format.

4.3.2 Internal Relations

On one hand, class `PhysModel` instances and internally *uses* class `NumEval` for initially calculating the reference value list, and for calculating mean square deviation values during the simulation. To enable class `NumEval` to parse expression with variables ("parameters"), is *provides* to the instance of class `NumEval` the required information about parameter values (see Chapter 4.2.6).

On the other hand, class `PhysModel` is instanced and managed by class `InductiveLawDiscoverer`, which is the top-level class of our simulation package (in its initial version), and connects the instances of `PhysModel` and `Farm`.

4.4 Top-Level Class InductiveLawDiscoverer

First of all, class `InductiveLawDiscoverer` provides the implementation for calculating the respective raw fitness values (function `.RawFitnessFunction(string)`). When instancing class `Farm`, it passes to the instance of `Farm` a function pointer to `.RawFitnessFunction(string)`, so that every time an individual needs to get an raw fitness value assigned, it can fetch this value from `InductiveLawDiscoverer`.

To be able to provide this value, `InductiveLawDiscoverer` also instances class `PhysModel`, thereby connecting all components.

4.4.1 Raw Fitness Calculation

In order to calculate the raw fitness of an expression, the `InductiveLawDiscoverer` class calls `PhysMode.GetResidualSquareSum()`, thereby calculating the mean quadratic deviation over all reference points by (under consideration of the tolerance levels for noisy values, see Chapter 4.3.1).

With vectorial expressions (e.g., the orbit of a planet in cartesian coordinates), several expressions are generated from the grammar (one for each dimension of the vector). The basis for the raw fitness is then simply the mean square deviation over all expressions.

Hence, the mean square deviation m for a D-dimensional expression $\mathrm{v}(d,t)$ that, for example, depends in each dimension d solely on time t, is calculated over the time interval $t_{min}...t_{max}$ with step width s as follows:

$$m = \frac{1}{n}\sum_{d=1}^{D}\sum_{j=0}^{(t_{max}-t_{min})/s}\left(\mathrm{v}_{\mathrm{ref}}(d, t_{min}+(t_{max}-t_{min})js) - \mathrm{v}\left(d, t_{min}+(t_{max}-t_{min})\,js\right)\right)^2 \quad (9)$$

where $\mathrm{v}_{\mathrm{ref}}(d,t)$ denotes the reference value for dimension d and at time t, and $\mathrm{v}(d,t)$ denotes the value of the to-be-tested function for dimension d and at time t.

Optionally, the following penalties are added to the mean square deviation:

- P_{const}: A configurable penalty value by which the mean square deviation value is increased if the expression is merely a constant; that is, if the parameters (e.g., time t or the path angle φ) do not occur in the generated expression. For multi-dimensional expressions, P_{const} is added, even if only one expression of the vector is a constant.
- $P_{complexity}$: A configurable penalty value that is multiplied with the expression's complexity as calculated by `NumEval.Complexity` (see Chapter 4.2.7).
- If the attempt to compute the numerical value of an expression does not result into a real value for all of the reference points (for example because at one point it results in a division by zero or because of the attempt to calculate the root of a negative number), then the "worst possible" fitness value $1.7 \cdot 10^{308}$ is assigned.

Hence, the raw fitness value f_i^{raw} for a D-dimensional expression that depends solely on the time t, is calculated as follows:

$$f_i^{raw} = \begin{cases} m + P_{const} + P_{complexity} \ldots m \in \mathbb{R} \\ 1.7 \cdot 10^{308} \ldots\ldots\ldots\ldots\ldots\ldots m \notin \mathbb{R} \end{cases} \quad (10)$$

4.4.2 Using the Whole Software Package via InductiveLawDiscoverer

The source code in Figure 7 gives an example on how to use the whole software package via the top-level class `InductiveLawDiscoverer`.

It is easy to see that this code example looks almost exactly like the example in Chapter 4.1. There are just two differences:

The `Farm` is now instanced by and accessed via the `InductiveLawDiscoverer` instance `ILD`, and - most importantly - using the functions described in the previous sub-chapters, it is now possible to define in lines 16-22 the to-be-discovered system (here: a simple system describing a unit cycle parametrized by the angle φ).

Additionally, in lines 24-27 it is demonstrated how to export the reference values into a file in CSV format for documentation purposes.

```
#include "InductiveLawDiscoverer.h"
#include "ga.h"
#define MAXDERIVDEPTH 70
#define SMALLESTSUBTREE 20
#define POPSIZE 10000
#define HISTORY 1
#define GPKRANDSEED -1
#define LANGUAGEDEF "c://temp/MyGrammar.def"
#define TARGETFITNESS 0.01

int main()

{
   InductiveLawDiscoverer ILD(POPSIZE, HISTORY, EXACTUNIF, LANGUAGEDEF,
                              MAXDERIVDEPTH, SMALLESTSUBTREE, GPKRANDSEED);
   ILD->PhysicalModel->SetReferenceFormula("Cos(phi):Sin(phi)");       // the "secret" formula
   ILD->PhysicalModel->AddParameter("phi", 0, 2*3.141, 0.1);           // value range
   ILD->PhysicalModel->AddNoiseParameter(0.0033, false, 0.01, false); // noise on dimension x
   ILD->PhysicalModel->AddNoiseParameter(0.0033, false, 0.01, false); // noise on dimension y
   ILD->PhysicalModel->CacheRefValues();                               // create ref values
   ILD->SetComplexityPenaltyWeight(1);                                 // complexity penalty
   ILD->SetMissingParamPenalty(5000);                                  // missing param penlty

   ofstream OutputFile;
   OutputFile.open("ref_value_table.csv");                             // write "noisy" ref
   OutputFile << ILD->PhysicalModel->GetRefValues();                   // values to file
   OutputFile.close();

   ILD->TheFarm.SetFitnessScaling(0, MINIMIZE);
   ILD->TheFarm.SetFitnessAdjustment(One_DividedBy_OnePlusX);
   ILD->TheFarm.SetSelection(LINEAR_RANK, 0.5);
   ILD->TheFarm.SetSampling(STOCHASTIC_UNIVERSAL, POPSIZE);
   ILD->TheFarm.SetMating(RANDOM_PERMUTATION);
   ILD->TheFarm.SetSelectionHeuristic(PropSelectionHeuristic, SEARCHSPACESIZE);
   ILD->TheFarm.SetCrossOver(1, 0.2);
   ILD->TheFarm.SetTreeMutProb(0.04);
   ILD->TheFarm.SetPointMutProb(0.03);
   ILD->TheFarm.SetSaveBest(YES);

   int i;
   i = 0;
   while(true)
   {
      i++;
      cout << endl << i << ": Fitness ";
      cout << ILD->TheFarm.Current->BestIndividual->RawFitness << ": ";
      ILD->TheFarm.Current->BestIndividual->Print();

      if(ILD->TheFarm.Current->BestIndividual->RawFitness<=TARGETFITNESS)
         break;

      ILD->TheFarm.NextStep()
   }
}
```

Figure 7: Sample source code.

5 First Round of GP Simulations

This chapter describes the first round of GP simulations, where the complete software package in its initial version (as described in the previous chapter) was used. Figure 8 shows the top-level source code as used in the first simulation round.

In Chapter 3.2, the problem of the (a priori unknown) best value for the maximum derivation depth was mentioned. This problem was taken into account by the following approach: In lines 17 and 19 of the program in Figure 8, a minimum and maximum value for the "maximum derivation depth" is defined. Line 18 defines the initial value. Lines 66-71 ensure that at each cycle of the main loop the current value of "maximum derivation depth" is increased by one, until the maximum value has been reached. Then, with each pass the current "maximum derivation depth" is decreased by one, until the minimum value is reached, after which it again starts to increase. By this, it is ensured that individuals of various derivation depths are considered in the process.

In Chapter 3.4 we mentioned that according to the author's experience the permanent inclusion of sub-trees in the selection and sampling process is not entirely beneficial for the overall performance. However, occasional inclusion seems to improve the performance. Therefore, lines 50, 51 and 72-76 were added. These lines of code include sub-trees to selection and sampling once every 10 steps.

```
#include "InductiveLawDiscoverer.h"
#include "ga.h"
#define HISTORY 1
#define GPKRANDSEED -1

int main()
{
   int i;
   int j;
   int minMDD;
   int tmpMDD;
   int maxMDD;
   int countMDD;
   int SmallestSubTreeOfVirtualPop;

   SmallestSubTreeOfVirtualPop = 0;
   minMDD = MIN_MDD;     // minimal "max. derivation depth"
   tmpMDD = START_MDD;   // current (initial) "max. derivation depth"
   maxMDD = MAX_MDD;     // maximum "max. derivation depth"
   countMDD = 1;         // if 1 count up, if -1 count down

   InductiveLawDiscoverer ILD(POPSIZE, HISTORY, EXACTUNIF, LANGUAGEDEF,
                              tmpMDD, SmallestSubTreeOfVirtualPop, GPKRANDSEED);

   ILD.PhysicalModel->SetReferenceFormula(SYS_FORMULA);
   ILD.PhysicalModel->AddParameter(MYPARAM, PARAM_FROM, PARAM_TO, STEP_WIDTH);
   ILD.PhysicalModel->CacheRefValues();
   ILD.SetComplexityPenaltyWeight(P_COMPLEX);
   ILD.SetMissingParamPenalty(P_PARAM);

   ILD.TheFarm->SetFitnessScaling(0, MINIMIZE);
   ILD.TheFarm->SetFitnessAdjustment(One_DividedBy_OnePlusX);
   ILD.TheFarm->SetSelection(LINEAR_RANK, TSR_MIN);
   ILD.TheFarm->SetSampling(STOCHASTIC_UNIVERSAL, POPSIZE);
   ILD.TheFarm->SetMating(RANDOM_PERMUTATION);
   ILD.TheFarm->SetSelectionHeuristic(PropSelectionHeuristic, SEARCHSPACESIZE);
   ILD.TheFarm->SetCrossOver(NO_OF_XOVER_POINTS, CROSSOVER_PROB);
   ILD.TheFarm->SetTreeMutProb(TREEMUT_PROB);
   ILD.TheFarm->SetPointMutProb(POINTMUT_PROB);
   ILD.TheFarm->SetSaveBest(YES);

   string ThisIndividual;
   string LastIndividual;
   OutputFile.open(FILENAME);
   OutputFile.precision(numeric_limits<double>::digits10 + 1);
   while(true)
   {
      i++;
      j++;
      if(j==10) // include sub-trees only once every 10 steps
         LDFarm->SetSmallestSubTreeOfVirtualPop(minMDD);

      ThisIndividual = ILD.TheFarm->Current->BestIndividual->PhenoTypeString();

      if (!(ThisIndividual.compare(LastIndividual) == 0))
      {
         // if best individual different than previously best: write to file
         OutputFile << endl << "Generation " << i << ": Fitness " <<
            ILD.TheFarm->Current->BestIndividual->RawFitness << ": " <<
            ThisIndividual.c_str() << endl;
         OutputFile.flush();
         LastIndividual = ThisIndividual;
      }

      if (tmpMDD >= maxMDD)
         countMDD = -1; // maximum "max. deriv depth" reached -> count down
      if (tmpMDD <= minMDD)
         countMDD = 1;  // minimum "max. deriv depth" reached -> count up
      tmpMDD = tmpMDD + countMDD;
      ILD.TheFarm->SetMaxDerivDepth(tmpMDD);

      if (j == 10)
      {   // de-activate inclusion of sub-trees for the next 10 steps
         ILD.TheFarm->SetSmallestSubTreeOfVirtualPop(0);
         j = 0;
      };

      ILD.TheFarm->NextStep();
   };
}
```

Figure 8: Top level source code used in simulation round one.

5.1 Force-Free Point-Like Particle

In this simulation the physical system represents the one-dimensional motion of a force-free point-like particle, as represented by the target formula

$$123 + 4.58 \cdot t \tag{11}$$

No random noise was superimposed on the simulated measurement values. Several simulations were carried out. The following sub-chapters give detailed information about one characteristic simulation run (internal identifier: GPK31C).

5.1.1 Pre-Simulations

Pre-simulations were executed with different simulation parameters to gain experience. Here are the results of these pre-simulations:

Expression	Corresponding Expanded Expression	Generation
$\sqrt{\pi}\left(\left(t + \frac{t}{\tanh\left(\frac{7.436}{10}\right)}\right) + 69.3953\right)$	$122.999949 + 4.580011 \cdot t$	67 811
$4.5801429 \cdot t + 122.999$	$122.999 + 4.5801429 \cdot t$	3 423
$\ln\left(8.22 + t\right) \cdot 57.18613$	-	6 776
$2.3 \cdot (54.434783 + t + t)$	$122.9 + 4.6 \cdot t$	275
$(t + 26.856) \cdot 4.5799851$	$123.000080 + 4.5799851 \cdot t$	937

Table 2: Results of various round-one pre-simulations for the “force-free point-like particle”-problem.

Seemingly, given grammar A and the described software in its initial version, only (albeit close) approximate solutions can be found for the “force-free point-like particle”-problem.

5.1.2 Reference Simulation Parameters

The following Table 3 lists all simulation parameters of the executed reference-simulation.

Simulation Parameter	Source-Code	Value
grammar	`LANGUAGEDEF`	Grammar A, see Appendix 13.3.1
formula	`SYS_FORMULA`	`123+4.58*t`
parameter	`MYPARAM`	t
range	`PARAM_FROM`, `PARAM_TO`	0 ... 10
step width	`STEP_WIDTH`	1
Population Size	`POPSIZE`	10000
minimum "max. deriv. depth"	`MIN_MDD`	20
maximum "max. deriv. depth"	`MAX_MDD`	50
initial "max. deriv. depth"	`START_MDD`	30, ascending
inclusion of sub-trees	lines 50, 51 and 72-76	min size 20 once every 10 steps
complexity penalty	`P_COMPLEX`	10
missing parameter penalty	`P_PARAM`	5000
selection	`LINEAR_RANK`	linear rank selection
minimum "target selection rate"	`TSR_MIN`	0.5
number of cross-over-points	`NO_OF_XOVER_POINTS`	1
cross-over probability	`CROSSOVER_PROB`	0.15
subtree mutation probability	`TREEMUT_PROB`	0.04
point-mutation probability	`POINTMUT_PROB`	0.02
elitism	`YES`	yes
fitness scaling	line 31	only over current population
fitness adjustment	`One_DividedBy_OnePlusX`	$1/(1+f_i^{std})$

Table 3: Simulation parameters for “force-free point-like particle”-problem, round one.

The used grammar ("Grammar A", see Appendix 13.3.1) is designed to produce expressions that are as directly human-readable as possible. Therefore, there are three different number definitions (`<integer>`, `<float>`, and `<float-sci>`), and each of these definitions has different variants to avoid leading pre-comma or trailing post-comma zeros.

5.1.3 Simulation Result

The generated protocol file can be found in Appendix 13.4.1 and lists the ever-improving best individuals. Figure 9 shows how the generated expressions converge towards the target function. Figure 10 shows the best result as found after 7023 generations (after which the simulation was stopped).

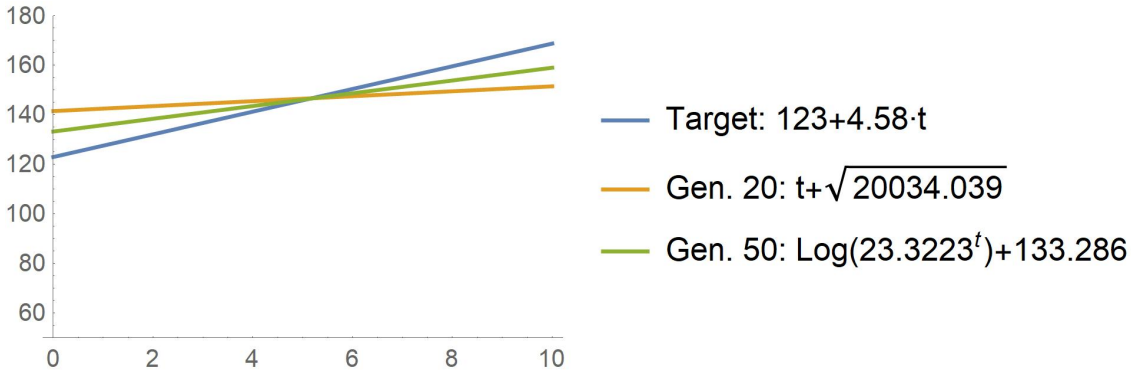

Figure 9: Round one - evolving expressions for problem "force-free point-like particle".

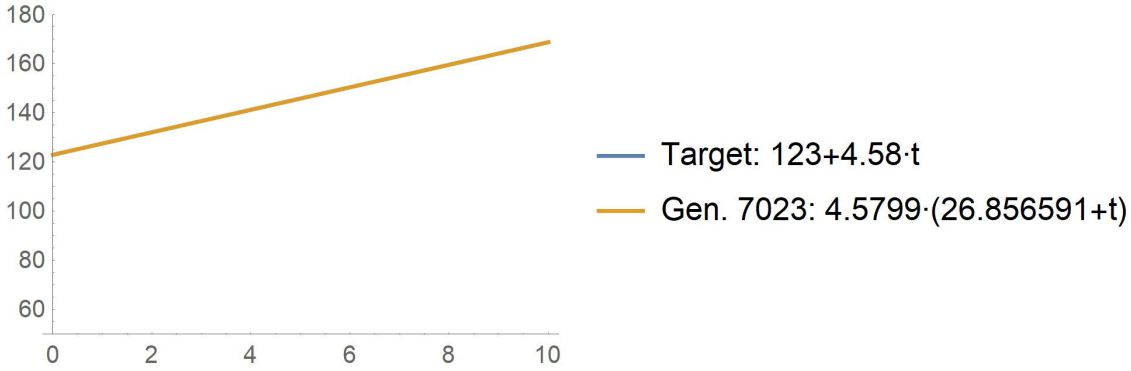

Figure 10: The approximate solution for the "force-free point-like particle"-problem found in round one

If we expand the term generated in generation 7023, we get $123.001 + 4.5799 \cdot t$. That is not exactly the target function $123 + 4.58 \cdot t$, but obviously a solution very close to it.

5.2 Accelerated Point-Like Particle in Homogeneous Gravity Field

In this simulation (internal identifier: GPK32J) the target function represents the height of a of a point-like particle, thrown from an elevated position, or, in short, the height of a ballistic trajectory. The target formula is

$$z(t) = 100 + 50t - \frac{9{,}80665}{2}t^2 = 100 + 50t - 4.903325t^2 \tag{12}$$

5.2.1 Pre-Simulations

In pre-simulations, when still using original grammar A, which also includes the `Min` and `Max` functions, the algorithm tended to produce piece-wise solutions such as the following:

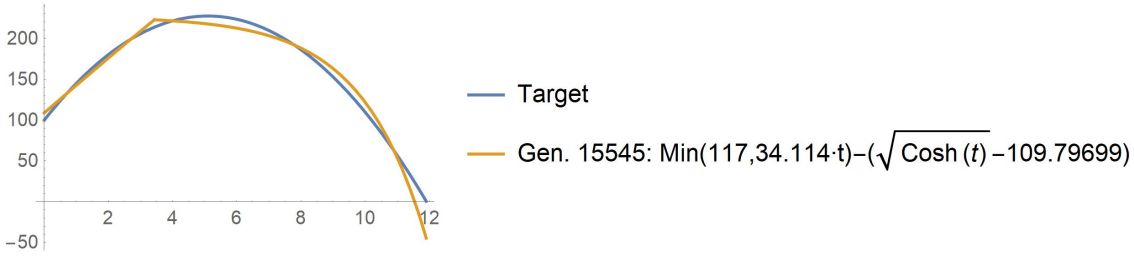

Figure 11: Round one - piece-wise approximate solution for the “ballistic trajectory”-problem.

Therefore, "Grammar B" (see Appendix 13.3.2) was used, which is almost equivalent to Grammar A, with the exception of functions `Min` and `Max`, which have been removed.

5.2.2 Simulation Parameters

Table 4 lists all relevant simulation parameters.

Simulation Parameter	Source-Code	Value
grammar	`LANGUAGEDEF`	Grammar B, see Appendix 13.3.2
formula	`SYS_FORMULA`	`100+50*t-9.80665/2*t^2`
parameter	`MYPARAM`	t
range	`PARAM_FROM`, `PARAM_TO`	0 ... 11.9
step width	`STEP_WIDTH`	0.1
Population Size	`POPSIZE`	10 000
minimum "max. deriv. depth"	`MIN_MDD`	20
maximum "max. deriv. depth"	`MAX_MDD`	50
initial "max. deriv. depth"	`START_MDD`	35, ascending
inclusion of sub-trees	lines 50, 51 and 72-76	min size 20 once every 10 steps
complexity penalty	`P_COMPLEX`	$\frac{i}{75}$ till $i < 100$, then Min$(100, \frac{i}{75} + \frac{i-100}{25})$
missing parameter penalty	`P_PARAM`	5000
selection	`LINEAR_RANK`	linear rank selection
minimum "target selection rate"	`TSR_MIN`	0.5
number of cross-over-points	`NO_OF_XOVER_POINTS`	1
cross-over probability	`CROSSOVER_PROB`	0.15
subtree mutation probability	`TREEMUT_PROB`	0.08
point-mutation probability	`POINTMUT_PROB`	0.05
elitism	`NO`	no
fitness scaling	line 31	only over current population
fitness adjustment	`One_DividedBy_OnePlusX`	$1/(1 + f_i^{std})$

Table 4: Simulation parameters for the “ballistic trajectory”-problem, round one.

5.2.3 Simulation Result

The generated protocol is listed in Appendix 13.4.2 and shows the evolving expressions. Please note that, in this simulation, until generation 2116 the *complexity penalty weight* $P_{complex}$ is dynamically increased. Therefore, until this point, the raw fitness value can increase, even if the expressions get actually better. Additionally, this simulation was carried out with the elitism operator deactivated. Hence, even after generation 2116 it is possible for the raw fitness value to grow occasionally.

Figure 12 shows how the generated expressions converge towards the target function. Figure 13 shows the best function found after 18317 generations (after which the simulation was stopped).

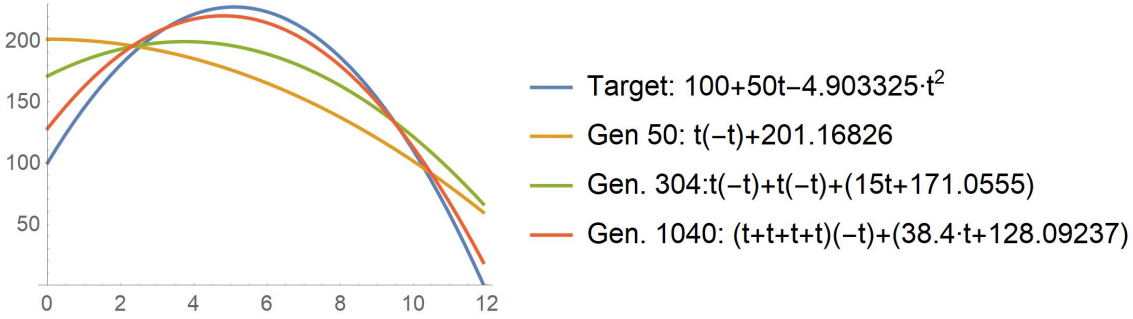

Figure 12: Round one - evolving expressions for the "ballistic trajectory"-problem.

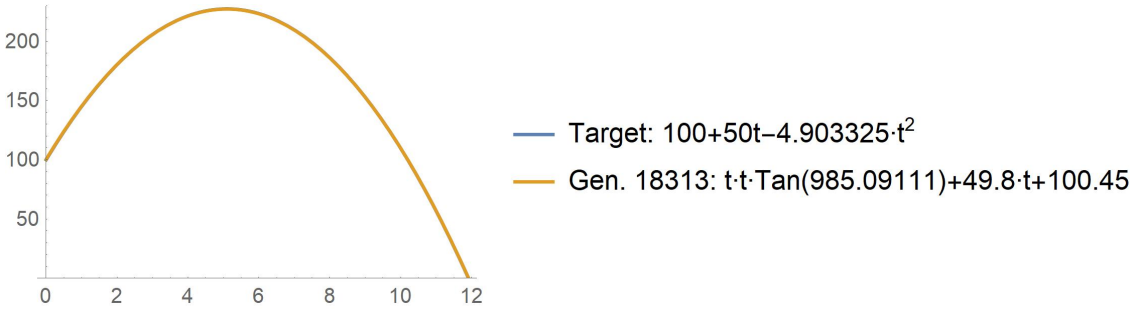

Figure 13: The approximate solution for the "ballistic trajectory"-problem found in round one.

If we expand the term generated in generation 18313, we get $100.45+49.8\cdot t-4.88763\cdot t^2$. That is not exactly the target function $100 + 50t - 4.903325t^2$, but again fairly close. The found solution corresponds in its structure with the target function (a second degree polynomial), however even in generation 18313 the constants are slightly deviating, and the constant of the quadratic term is expressed in a cumbersome way (tan(985.09111)).

5.3 Path of a Satellite Around a Central Gravitational Source in 1D Polar Coordinates

In this simulation (internal identifier: GPK33A) the target function represents the elliptical path of a satellite around a central gravitational source in one-dimensional polar coordinates, parameterized by the path angle φ:

$$\mathrm{r}(\varphi) = \frac{1}{\varepsilon \cdot \cos(\varphi)} \text{ with } \varepsilon \stackrel{!}{=} \frac{\sqrt{5^2 - 3^2}}{5} = 0.8 \tag{13}$$

5.3.1 Simulation Parameters

Table 5 lists all relevant simulation parameters.

Simulation Parameter	**Source-Code**	**Value**
grammar	`LANGUAGEDEF`	Grammar C, see Appendix 13.3.3
formula	`SYS_FORMULA`	`1/(1+0.8*Cos(phi)))`
parameter	`MYPARAM`	phi
range	`PARAM_FROM`, `PARAM_TO`	0 ... 2*3.141
step width	`STEP_WIDTH`	0.1
Population Size	`POPSIZE`	100 000
minimum "max. deriv. depth"	`MIN_MDD`	20
maximum "max. deriv. depth"	`MAX_MDD`	70
initial "max. deriv. depth"	`START_MDD`	40, ascending
inclusion of sub-trees	lines 50, 51 and 72-76	min size 20 once every 10 steps
complexity penalty	`P_COMPLEX`	0.0001
missing parameter penalty	`P_PARAM`	5000
selection	`LINEAR_RANK`	linear rank selection
minimum "target selection rate"	`TSR_MIN`	0.5
number of cross-over-points	`NO_OF_XOVER_POINTS`	1
cross-over probability	`CROSSOVER_PROB`	0.15
subtree mutation probability	`TREEMUT_PROB`	0.04
point-mutation probability	`POINTMUT_PROB`	0.02
elitism	`YES`	yes
fitness scaling	line 31	only over current population
fitness adjustment	`One_DividedBy_OnePlusX`	$1/(1 + f_i^{std})$

Table 5: Simulation parameters for the “elliptic satellite path in 1D polar coordinates”-problem, round one.

This time, a significantly larger population size of 100,000 individuals was chosen.

Grammar C is equally as powerful as grammar A. There are just two formal differences:

- The non-terminal symbol `<parameter>` is now defined as `phi` (as the function depends on the angle, and not on the time); and
- the pseudo-function `Pi()` is now an expansion of `<number_pos>`, as it is not really a function, but rather a numerical constant.

5.3.2 Simulation Result

The generated protocol is listed in Appendix 13.4.3 and shows the evolution process.

Figure 14 shows how the generated expressions converge towards the target function. Figure 15 shows the best function found after 197 generations (after which the simulation was stopped).

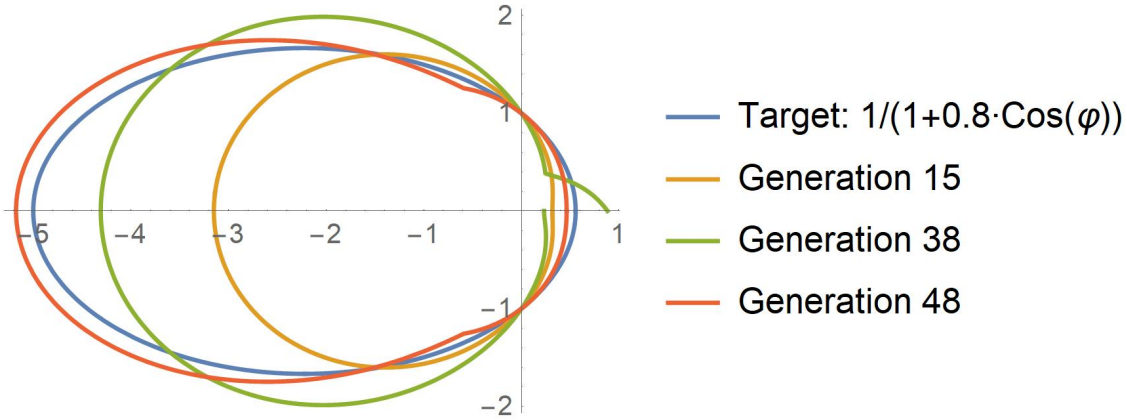

Figure 14: Round one - evolving expressions for the “Elliptic Satellite Path in 1D Polar Coordinates”-problem.

These are the evolving expressions as shown in Figure 14:

Expression	Generation
$\arcsin(\cos(\sqrt{\arctan(4\,415\,609.2)}))^{\cos(\varphi)}$	15
$\operatorname{arcsinh}\left(\cos\left(\sqrt{\min\left(\sinh(1.341), \frac{1.7741342}{\tanh(465,4\cdot 920.67224)}\right)}\right)\right)^{\cos(\varphi)}$	38
$\arcsin\left(\cos\left(\arctan\left(\max\left(\tan(431\,580.54), \tan(\cos(4776.5972))\right)^{\cos(\varphi)}\right)\right)\right)^{\cos(\varphi)}$	48

Table 6: Evolving expressions as shown in Figure 14.

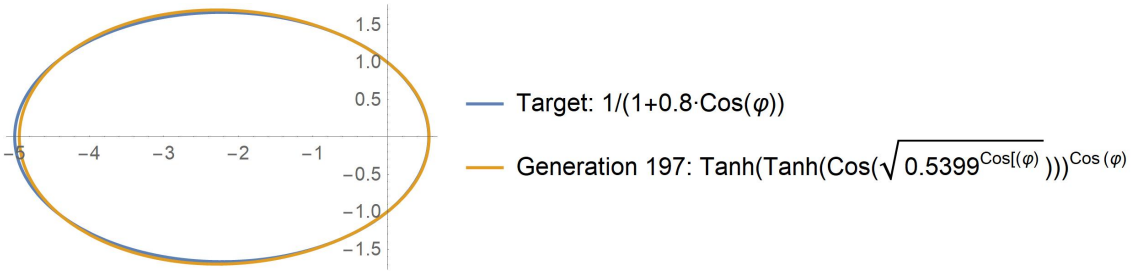

Figure 15: The approximate solution for the "Elliptic Satellite Path in 1D Polar Coordinates"-problem found in round one.

Although the found approximation does not look very similar to the target function, both can be compared by means of a power series development. Developing the target function into a power series about the point $\varphi = 0$, results in:

$$0.555556 + 0.123457\varphi^2 + 0.0171468\varphi^4 + 0.00186709\varphi^6 + 0.000167461\varphi^8 + O\left(\varphi^9\right) \quad (14)$$

A comparison with the power series of the found approximate solution reveals the similarity:

$$0.558286 + 0.131219\varphi^2 + 0.0126324\varphi^4 + 0.00185249\varphi^6 + 0.000308289\varphi^8 + O\left(\varphi^9\right) \quad (15)$$

6 Conclusions After First Round and Consequently Introduced Software Improvements

6.1 Conclusions

In the first simulation round, the GP simulations could always only find approximate solutions. This was mainly for the following two reasons:

- **Population Size:** Although population sizes of about 10,000 (as have been used in Chapters 5.1 and 5.2) are deemed sufficient for simpler problems (see [Poli et al., 2008, p. 26])), symbolic regression problems of the given complexity usually require population sizes of 100,000 or more (see e.g. [Poli et al., 2008, p. 163]).
- **Software Time Performance**: However, with a population size of 100,000, as has been, for example, used in Chapter 5.3, the initial version of our software already becomes painstakingly slow and definitely has reached its performance limit. The reason for this is that it does not use multi-threading techniques to utilize the multiple cores of modern processors.

- **Premature Convergence:** Premature convergence is a well-discussed challenge in GP. The method of choice to overcome this problem is to split up the total population into multiple gene-pools, in each of which the evolution process can run independently for most of the time (see e.g. [Poli et al., 2008, p. 83ff]). An exchange of individuals should occur only occasionally in a very limited manner. This idea matches very well with the goal to enable the software for multi-threading.

6.2 Implemented Software Improvements

6.2.1 Multi-Threaded Raw Fitness Calculation

An option was added in class `InductiveLawDiscoverer` for allowing multi-threaded raw-fitness calculations even with conventional singe-gene-pool simulations as described in Chapter 5.

6.2.2 Multi-Threaded Parallel Gene Pools

More importantly, a new top-level class `LawDiscovererFarm` was developed. This class creates a configurable number of `InductiveLawDiscoverer`-classes in parallel threads, and runs the according simulations in parallel.

These simulations are initially independent of each other. Only after a configurable number of steps, it is decided at each subsequent step at random (with a configurable probability) whether individuals are exchanged between the sub-simulations. In such case, the best individual is copied from a randomly chosen simulation ("Farm") into a randomly selected other sub-simulation. It is also possible to determine how many such exchanges should be carried out in this step. Thereafter, the exchange of individuals is blocked again for the pre-configured number of steps.

6.2.3 Summary

The new implementation, as shown in Figure 16 (with the new class highlighted in orange), has the following significant advantages:

- While the initial software version was able to handle population sizes of typically 10,000 to at most 100,000 with reasonable performance, up to 250 sub-simulations with between 2,500 and 10,000 individuals can be handled with the improved software at a reasonable speed. This is a tenfold increase in capacity to up to around 1,000,000 (or even more) individuals.
- As discussed in Chapter 6.1, establishing independent gene-pools also provides a way out of the problem of the algorithm often remaining stuck in a local optimum. In many parallel sub-simulations it is more likely that different optimization paths are found.

Figure 17 shows the new top-level source code, utilizing the new features.

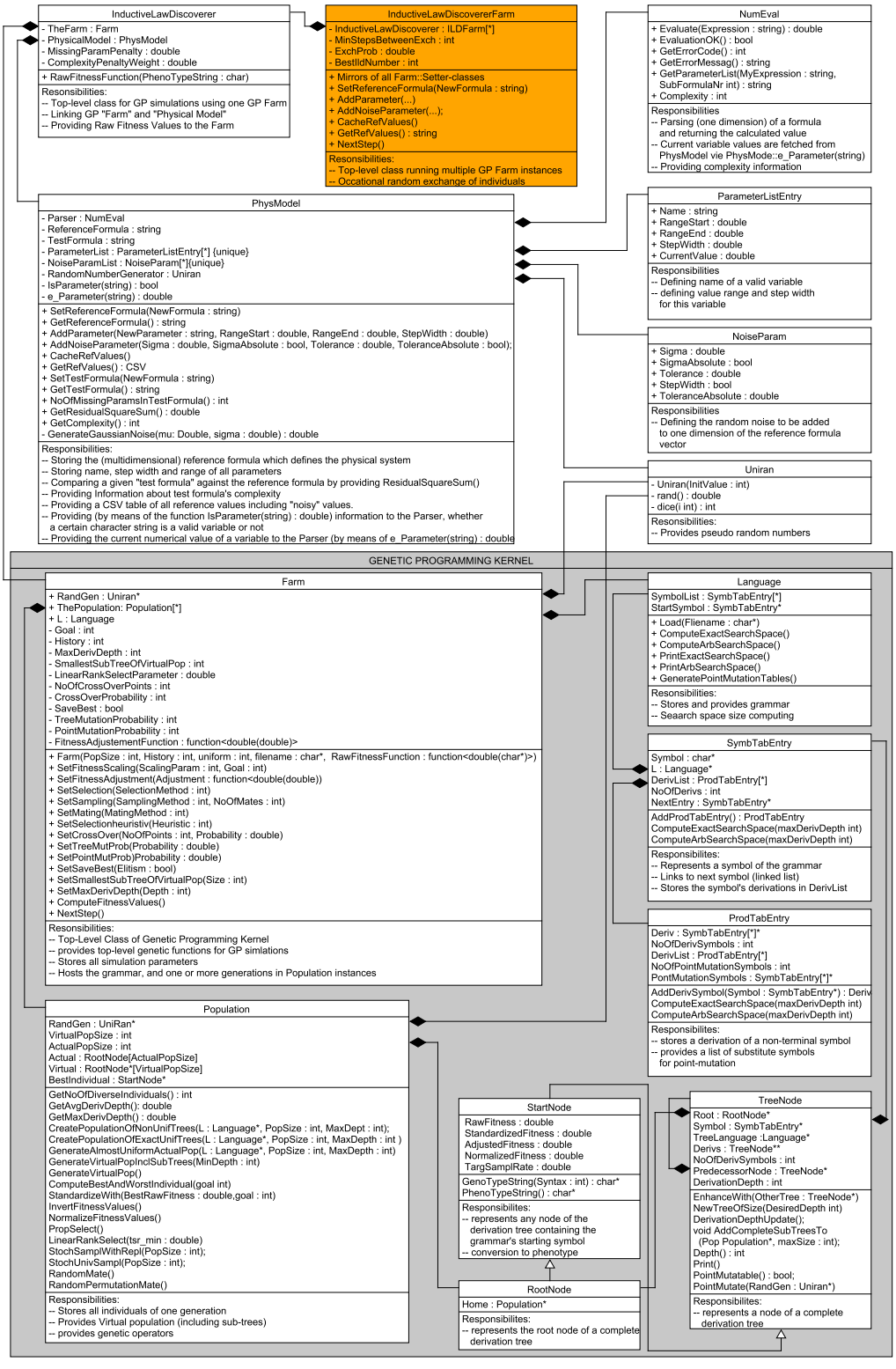

Figure 16: UML Diagram of improved software package, now including a new top-level class (highlighted in orange) allowing multi-threaded parallel sub-simulations.

```
#include "InductiveLawDiscoverer.h"
#include "ga.h"
#define HISTORY 1

int main()
{
   int i;
   int j;
   int minMDD;
   int tmpMDD;
   int maxMDD;
   int countMDD;
   int SmallestSubTreeOfVirtualPop;

   SmallestSubTreeOfVirtualPop = 0;
   minMDD = MIN_MDD;     // minimal "max. derivation depth"
   tmpMDD = START_MDD;   // current (initial) "max. derivation depth"
   maxMDD = MAX_MDD;     // maximum "max. derivation depth"
   countMDD = 1;         // if 1 count up, if -1 count down

   LawDiscovererFarm ILDF(GENEPOOLCOUNT, POPSIZE, HISTORY, EXACTUNIF, LANGUAGEDEF,
                          tmpMDD, SmallestSubTreeOfVirtualPop);

   ILDF.PhysicalModel->SetReferenceFormula(SYS_FORMULA);
   ILDF.PhysicalModel->AddParameter(MYPARAM, PARAM_FROM, PARAM_TO, STEP_WIDTH);
   ILDF.ExchProb = EXCH_PROB;
   ILDF.MinStepsBetweenExch = MIN_STEPS_BE;
   ILDF.NoOfExchgs = NO_OF_EXCHG;
   ILDF.PhysicalModel->CacheRefValues();
   ILDF.SetComplexityPenaltyWeight(P_COMPLEX);
   ILDF.SetMissingParamPenalty(P_PARAM);
   ILDF.SetFitnessScaling(0, MINIMIZE);
   ILDF.SetFitnessAdjustment(One_DividedBy_OnePlusX);
   ILDF.SetSelection(LINEAR_RANK, TSR_MIN);
   ILDF.SetSampling(STOCHASTIC_UNIVERSAL, POPSIZE);
   ILDF.SetMating(RANDOM_PERMUTATION);
   ILDF.SetSelectionHeuristic(PropSelectionHeuristic, SEARCHSPACESIZE);
   ILDF.SetCrossOver(NO_OF_XOVER_POINTS, CROSSOVER_PROB);
   ILDF.SetTreeMutProb(TREEMUT_PROB);
   ILDF.SetPointMutProb(POINTMUT_PROB);
   ILDF.SetSaveBest(YES);
   ILDF.ComputeFitnessValues(YES);

   string ThisIndividual;
   string LastIndividual;
   OutputFile.open(FILENAME);
   OutputFile.precision(numeric_limits<double>::digits10 + 1);
   while(true)
   {
      i++;
      j++;
      if(j==10) // include sub-trees only once every 10 steps
         ILDF.SetSmallestSubTreeOfVirtualPop(minMDD);

      ThisIndividual = ILDF.BestIndividual->PhenoTypeString();

      if (!(ThisIndividual.compare(LastIndividual) == 0))
      {
         // if best individual different than previously best: write to file
         OutputFile << endl << "Generation " << i << ", Farm " << ILDF.BestIDLNumber;
         OutputFile << ", MDD " << tmpMDD;
         OutputFile << ", Fitness " << ILDF.BestIndividual->RawFitness << ": " << endl;
         OutputFile << ThisIndividual.c_str() << endl;
         OutputFile.flush();
         LastIndividual = ThisIndividual;
      }

      if (tmpMDD >= maxMDD)
         countMDD = -1; // maximum "max. deriv depth" reached -> count down
      if (tmpMDD <= minMDD)
         countMDD = 1;  // minimum "max. deriv depth" reached -> count up
      tmpMDD = tmpMDD + countMDD;
      ILDF.SetMaxDerivDepth(tmpMDD);

      if (j == 10)
      {   // de-activate inclusion of sub-trees for the next 10 steps
         ILDF.SetSmallestSubTreeOfVirtualPop(0);
         j = 0;
      };

      ILDF.NextStep();
   };
}
```

Figure 17: Source code of main loop as used in second simulation round.

7 Second Round of GP Simulations

This chapter describes the second round of GP simulations, where the software package in its improved version (multi-threading-enabled and allowing for multiple gene-pools, as described in Chapter 6.2), was used. In Figure 17 the top-level source code used in this second simulation round is shown.

A comparison with the previous main loop source code (as shown in Figure 8) reveals that the overall structure has not changed. The main difference is that everything is now accessed via the single `LawDiscovererFarm`-instance `ILDF`, and while instancing `ILDF` in line 21, the number of parallel running gene-pools can be specified (`GENEPOOLCOUNT`). The total number of individuals in the simulation is then `GENEPOOLCOUNT` multiplied with `POPSIZE`.

7.1 Force-Free Point-Like Particle

In this simulation (internal simulation identifier: GPK31H), the same physical system as in Chapter 5.1 (one-dimensional motion of a force-free point-like particle), was simulated. Again, no random noise was superimposed on the simulated measurement values.

7.1.1 Simulation Parameters

Table 7 shows all simulation parameters.

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar B, see Appendix 13.3.2
formula	SYS_FORMULA	123+4.58*t
parameter	MYPARAM	t
range	PARAM_FROM, PARAM_TO	0 ... 10
step width	STEP_WIDTH	1
no. of parallel gene-pools	GENEPOOLCOUNT	50
population size per pool	POPSIZE	5 000
total no. of individuals	GENEPOOLCOUNT*POPSIZE	250 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	17
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	20
maximum "max. deriv. depth"	MAX_MDD	50
initial "max. deriv. depth"	START_MDD	30, ascending
inclusion of sub-trees	lines 52, 53 and 75-79	min size 20 once every 10 steps
complexity penalty	P_COMPLEX	10
missing parameter penalty	P_PARAM	5 000
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.15
subtree mutation probability	TREEMUT_PROB	0.04
point-mutation probability	POINTMUT_PROB	0.02
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1+f_i^{std})$

Table 7: Simulation parameters for “force-free point-like particle”-problem, round two.

7.1.2 Simulation Result

The generated protocol file is listed in Appendix 13.4.4. The solution $\mathbf{123 + 4.58 \cdot t}$ (found in generation 1347) **exactly matches the target function**, which is a significant improvement.

7.2 Accelerated Point-Like Particle in Homogeneous Gravity Field

In this simulation (internal simulation identifier: GPK32O), the same physical system as in Chapter 5.2 (accelerated point-like particle in homogeneous gravity field), was simulated. Again, no random noise was superimposed on the simulated measurement values.

7.2.1 Simulation Parameters

Table 8 shows all simulation parameters.

Simulation Parameter	**Source-Code**	**Value**
grammar	`LANGUAGEDEF`	Grammar A, see Appendix 13.3.1
formula	`SYS_FORMULA`	`100+50*t-9.80665/2*t^2`
parameter	`MYPARAM`	t
range	`PARAM_FROM`, `PARAM_TO`	-50 ... 50
step width	`STEP_WIDTH`	1
no. of parallel gene-pools	`GENEPOOLCOUNT`	90
population size per pool	`POPSIZE`	7 500
total no. of individuals	`GENEPOOLCOUNT*POPSIZE`	675 000
individual exchange probability	`EXCH_PROB`	0.2
min. steps between individual exchg.	`MIN_STEPS_BE`	17
no. of exchanged individuals	`NO_OF_EXCHG`	1
minimum "max. deriv. depth"	`MIN_MDD`	20
maximum "max. deriv. depth"	`MAX_MDD`	50
initial "max. deriv. depth"	`START_MDD`	30, ascending
inclusion of sub-trees	lines 52, 53 and 75-79	min size 20 once every 10 steps
complexity penalty	`P_COMPLEX`	10
missing parameter penalty	`P_PARAM`	5 000
selection	`LINEAR_RANK`	linear rank selection
minimum "target selection rate"	`TSR_MIN`	0.5
number of cross-over-points	`NO_OF_XOVER_POINTS`	1
cross-over probability	`CROSSOVER_PROB`	0.15
subtree mutation probability	`TREEMUT_PROB`	0.04
point-mutation probability	`POINTMUT_PROB`	0.02
elitism	`YES`	yes
fitness scaling	line 31	only over current population
fitness adjustment	`One_DividedBy_OnePlusX`	$1/(1+f_i^{std})$

Table 8: Simulation parameters for “accelerated point-like particle in homogeneous gravity field”-problem, round two.

Although the physically relevant parameter range only goes from $t_{min} = 0$ to $t_{max} = 11.9$, the simulation range was chosen intentionally to go from $t_{min} = -50$ to $t_{max} = +50$. This is to prevent the simulation from finding solutions that are only local approximations in the physically relevant range.

7.2.2 Simulation Result

The generated protocol file is listed in Appendix 13.4.5. The solution found in generation 4468 is a very good approximation to the target function $100 + 50t - 4.903325 \cdot t^2$:

$$99.9999 - t \cdot (4.9033249 \cdot t - 50) \tag{16}$$

Although there are again small deviations in the number constants, the solution is significantly better than the approximate solution generated in the first round (Chapter 5.2).

The advantage of parallel sub-simulations can be seen well in this simulation. Between generations 1275 and 1693, the sub-simulation in "Farm 44", in which a constant is expressed as "cos(77.174404)", dominates. However, in the background the sub-simulation in "Farm 13", which eventually does not require such "function constants", silently develops. From generation 1883 on, the best individual in "Farm 13" has the better fitness values (because of the complexity penalty value), and ultimately leads to the final solution.

Also, grammar A can now be used again in this simulation, without the Min and Max functions affecting the result.

7.3 Path of a Satellite Around a Central Gravitational Source in 1D Polar Coordinates

In this simulation (internal simulation identifier: GPK33D), the same physical system as in Chapter 5.3 ("elliptic satellite path in 1D polar coordinates"), was simulated. Again, no random noise was superimposed on the simulated measurement values.

7.3.1 Simulation Parameters

Table 9 shows all simulation parameters.

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar C, see Appendix 13.3.3
formula	SYS_FORMULA	1/(1+0.8*Cos(phi)))
parameter	MYPARAM	phi
range	PARAM_FROM, PARAM_TO	0 ... 4*3.141
step width	STEP_WIDTH	0.2
no. of parallel gene-pools	GENEPOOLCOUNT	50
population size per pool	POPSIZE	5 000
total no. of individuals	GENEPOOLCOUNT*POPSIZE	250 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	17
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	20
maximum "max. deriv. depth"	MAX_MDD	50
initial "max. deriv. depth"	START_MDD	30, ascending
inclusion of sub-trees	lines 52, 53 and 75-79	min size 20 once every 10 steps
complexity penalty	P_COMPLEX	0.005
missing parameter penalty	P_PARAM	5 000
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.15
subtree mutation probability	TREEMUT_PROB	0.04
point-mutation probability	POINTMUT_PROB	0.02
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1+f_i^{std})$

Table 9: Simulation parameters for "elliptic satellite path in 1D polar coordinates"-problem, round two.

The parameter range was chosen to run from $\varphi_{min} = 0$ to $\varphi_{max} = 4\pi$. This is to favor solutions that do not "spiral" in or out with increasing value of φ.

7.3.2 Simulation Result

The generated protocol file is listed in Appendix 13.4.6. The solution found in generation 6488

$$\frac{1.25}{\cos(\varphi) + 1.25} \tag{17}$$

exactly matches the target function $1/(1 + 0.8 \cdot \cos(\varphi))$, if numerator and denominator are expanded by 1.25. This is a significant improvement compared to the best first-round simulation.

7.4 Path of a Satellite Around a Central Gravitational Source in 2D Cartesian Coordinates

In this simulation (internal identifier: GPK34A) the target function represents the elliptical path of a satellite around a central gravitational source in two-dimensional Cartesian coordinates, parameterized by the path angle φ:

$$\vec{\mathrm{r}}(\varphi) = \begin{pmatrix} \frac{p\cos\varphi}{\varepsilon\cdot\cos(\varphi)} \\ \frac{p\sin\varphi}{\varepsilon\cdot\cos(\varphi)} \end{pmatrix} \quad \text{with} \quad p \stackrel{!}{=} 1 \;\text{ and }\; \varepsilon \stackrel{!}{=} \frac{\sqrt{5^2 - 3^2}}{5} = 0.8 \tag{18}$$

7.4.1 Simulation Parameters

Table 10 shows all simulation parameters.

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar D, see Appendix 13.3.4
formula	SYS_FORMULA	Cos(phi)/(1+0.8*Cos(phi)): Sin(phi)/(1+0.8*Cos(phi))
parameter	MYPARAM	phi
range	PARAM_FROM, PARAM_TO	0 ... 4*3.141
step width	STEP_WIDTH	1
no. of parallel gene-pools	GENEPOOLCOUNT	50
population size per pool	POPSIZE	5 000
total no. of individuals	GENEPOOLCOUNT*POPSIZE	250 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	17
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	50
maximum "max. deriv. depth"	MAX_MDD	120
initial "max. deriv. depth"	START_MDD	60, ascending
inclusion of sub-trees	lines 52, 53 and 75-79	min size 40 once every 10 steps
complexity penalty	P_COMPLEX	0.005
missing parameter penalty	P_PARAM	5 000
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.15
subtree mutation probability	TREEMUT_PROB	0.04
point-mutation probability	POINTMUT_PROB	0.02
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1+f_i^{std})$

Table 10: Simulation parameters for “elliptic satellite path in 2D Cartesian coordinates”-problem, round two.

Grammar D is equal to Grammar C, except that <expression> now represents a two-dimensional vector. The parameter range was chosen to run from $\varphi_{min} = 0$ to $\varphi_{max} = 4\pi$, to favor solutions that do not “spiral” in or out.

7.4.2 Simulation Result

The generated protocol is listed in Appendix 13.4.7 and shows the evolution process.

Figure 18 shows how the generated expressions converge towards the target function. Figure 19 shows the best function found after 267 generations (after which the simulation was stopped).

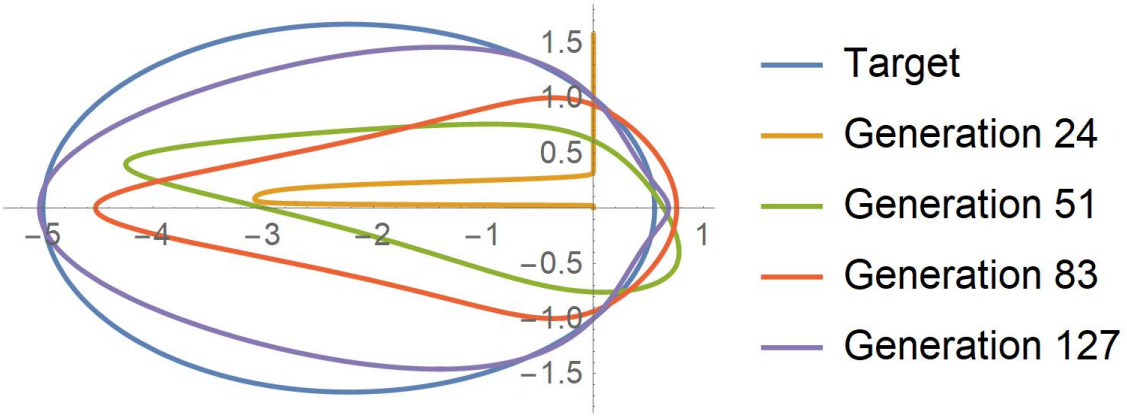

Figure 18: Round two - evolving expressions for the “elliptic satellite path in 2D Cartesian coordinates”-problem.

These are the evolving expressions as shown in Figure 18:

Expression	Generation
$\begin{pmatrix} -\pi \left(e^{\sinh(-1.20475-3.29935)}\right)^{2180.8^{\cos(\varphi)}} \\ \arccos(\tanh(\varphi)) \end{pmatrix}$	24
$\begin{pmatrix} \tan(4.231+\cos\left(\varphi-e^{\ln(119.317)}\right)+4.2-(\sin\left(\sinh(-119.117)+\operatorname{arcsinh}(\varphi)\right))) \\ \tanh(\cos(\varphi-e^{\ln(8.223333)})) \end{pmatrix}$	51
$\begin{pmatrix} \tan(\cos(\varphi)+5.93) \\ \sin(\varphi) \end{pmatrix}$	83
$\begin{pmatrix} \frac{\cos(\varphi)}{\cos(\cos(\varphi))} \\ e^{\cos(-\varphi)} \\ \frac{\sin(\varphi)}{e^{\cos(\varphi)}} \end{pmatrix}$	127

Table 11: Evolving expressions as shown in Figure 18.

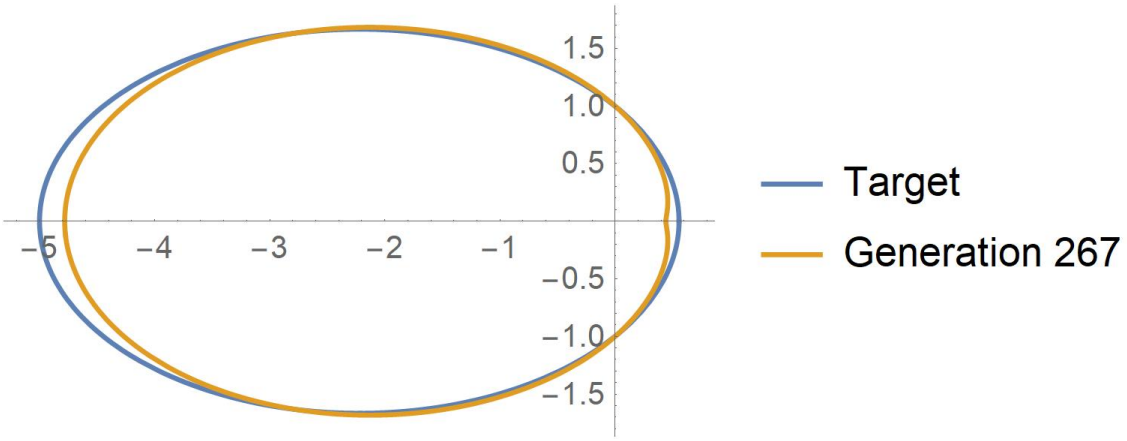

Figure 19: The approximate solution for the "Elliptic Satellite Path in 2D Cartesian Coordinates"-problem found in round two.

This is the approximate solution as found in generation 267, and as depicted in Figure 19:

$$\vec{r}(\varphi) = \begin{pmatrix} \frac{\cos\varphi}{\operatorname{arcsinh}(e^{\tan(\cos(\varphi))})} \\ \frac{\sin\varphi}{e^{\tan(\cos(\varphi))}} \end{pmatrix} \tag{19}$$

8 Conclusions After Second Round and Consequently Introduced Software Improvements

8.1 Conclusions

With the improved software version, the GP simulations performed significantly better. Whereas in the first round the software was never able to come up with an exact solution, the improved software found the exact solution both for the "force-free point-like particle"-problem (Chapters 5.1 and 7.1), as well as for the "elliptic satellite path in 1D polar coordinates"-problem (Chapter 5.3 and 7.3). For the ballistic problem (Chapters 5.2 and 7.2), a much better-than-before approximate solution could be found.

However, a look into the second-round third simulation protocol (listed in Chapter 13.4.6) reveals that it took the software almost 5000 generations to jump from 1.2499999 to 1.25 in the denominator expression; and it was mere luck that the same number was required in the numerator, so that the crossover-operator was able to do the final bit. Obviously, the algorithm has a problem in the "end-game".

These are the identified reasons:

- **Grammars optimized to produce "nice" numbers:** All the grammars used so far express numbers in a very complicated way, as to ensure that they are as human-readable as possible. For example, it is ensured that there are no leading or trailing zeros, and there are separate instances for integers, floats and numbers in scientific notation. The drawback of this is that with such a grammar (where 001.3000000 is not allowed) it is very difficult to jump from 1.2499999 to 1.5, or from 99.999999 to 100, or from 1.234 to 1.2345, as always a complete new number must be randomly generated.
- **Decimal number representation:** But even if we allowed for number representations like 1.3000000, it still would be impossible to reach 1.3000000 from 1.2499999 by point-mutation. The reason is that the minuscule step of 0.0000001 requires the modification of 7 digits simultaneously. That is not even theoretically possible, because both the initial and also the improved version of the software always performs just a single point mutation on an individual at a time. And, if we allowed for multiple point-mutations simultaneously, or just depend on the tree-mutation operator, randomly generating the correct new number in the course of the evolutionary process is still practically impossible, as the probability for this to happen is so extremely small.

8.2 Implemented Software Improvements

8.2.1 10-ary Gray Encoding

Some researchers in the GA-domain (e.g. [Caruana et al., 1988]) have therefore suggested representing numbers in Gray encoding instead of decimal or usual binary encoding. Originally, Gray encoding meant a binary numeral system where two successive values differ in only one bit, and was invented (and patented) by Frank Gray [Gray, 1953] to solve a pulse code communication problem.

Later, the concept was extended and generalized to the q-ary Gray code. In contrast to the original Gray code that works on binary numbers, a q-ary Gray code uses a number system of base q (see [Mambou et al., 2016]). To include gray-coding, the GPK software (and, specifically the `NumEval`-class) was extended to be able to decode 10-ary Gray-Coded numbers. The implementation is based on the algorithm described in [Mambou et al., 2016, p. 381].

In the updated software version, the parser interprets any string of digits preceded by the sequence `&G` as a 10-ary Gray-coded number. Leading zeros are allowed. If the number is followed by an `E`, optionally followed by `+` or `-`, and then followed by one or two further digits, these digits will be interpreted as the exponent in 10-ary Gray notation.

8.2.2 Extended Grammar Definition Allowing For Non-Terminal Symbol Default Values

With the new parser able to interpret 10-ary Gray-coded numbers, a matching grammar definition must be found. 10-ary Gray-coded numbers could be, for example, defined as follows:

```
<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;
<number_pos> := "&G" <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08";
```

This example would allow for the generation of 10-ary Gray-coded numbers that translate to a decimal number range of 0000000000.00000000 to 9999999999.99999999. Of course, with such definition, 99% of the randomly generated numbers are larger than 10000000. This means that small constants almost never just "pop up". They only could be generated by mutation from larger numbers, which takes a lot of time.

To overcome this problem, we could try to modify the grammar, to look like this:

```
<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;
<number_pos> := "&G" <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" "0" <digit> <digit> <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" "0" "0" <digit> <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" "0" "0" "0" <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" "0" "0" "0" "0" <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" "0" "0" "0" "0" "0" <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" "0" "0" "0" "0" "0" "0" <digit> <digit> ... <digit> "E-08" |
                "&G" "0" "0" "0" "0" "0" "0" "0" <digit> ... <digit> "E-08" |
                "&G" "0" "0" "0" "0" "0" "0" "0" "0" ... <digit> "E-08" |
                ...
```

This definition allows the random generation of numbers in different number ranges. Although this seems to be a good idea at first glance, a little contemplation reveals a serious flaw: The leading zeroes are intended to allow for mutation towards larger numbers. However, these zeroes can actually never mutate to any other digit, because they are not at all related to the `<digit>` symbol.

As a solution to this, we enhanced the syntax of the grammar definition file and implemented the possibility of non-terminal symbols with a default value. This default value must, of course, be one of the possible substitution symbols. It forces the GPK to *always* use this default symbol when generating a new random derivation tree.

The syntax for a `<digit>` symbol with default value `"0"` is:

```
<digit:0>
```

And with this enhancement, it now becomes possible to actually define fully mutatable 10-ary gray-coded numbers with different initial number ranges:

```
<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;
<number_pos> := "&G" <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> ... <digit> "E-08" |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> ... <digit> "E-08" |
                ...
```

8.2.3 Multiple Application of Cross-Over Operator and Mutation Operators

To further improve performance, a new feature that allows the definition of separate repetition probabilities for the cross-over operator, the tree-mutation operator and the point-mutation operator was implemented. These repetition probabilities define the probability with which the respective operator is repeated after its first application.

So, for example, if we define the point-mutation probability to be 0.04, and the respective repetition probability to be 0.5, this means:

- The probability for every individual to be subjected to the point-mutation operator at all is 4 %.
- If an individual is selected for point-mutation, then there is always at least one point mutation.
- After that, there is a 50% chance for a second point mutation. Therefore, in our example an individual has a chance of 2% to be point-mutated twice in a row.
- After that, there is again a 50% chance for a third point mutation. Therefore, in our example an individual has a chance of 1% to be point-mutated three times in a row.
- Etc.

8.2.4 Numerical "Endgame" Improvements

- It can happen that the algorithm has found an *almost* exact solution for a problem with a complexity of, let's say, 13; and another *really perfect* solution with a complexity of 14. If the complexity penalty is, let's say, 1, and the mean square deviation of the inferior individual is smaller than 1, then the better solution will never become visible. To prevent this from happening, a *Bull's-Eye Bonus Factor* can be defined, by which the complexity penalty will be multiplied if an individual has zero deviation to the target function. So, a *Bull's-Eye Bonus Factor* of 0.8 would cause in our example the better individual to now get a complexity penalty of just $14 \cdot 0.8 = 11.2$ assigned, thereby outranking the less-than-perfect individual.
- It is one of the subtleties of numerical computation that symbolically equivalent expressions (like `"t*t"` and `"t^2"`) can lead to slightly deviating numerical values. In such case (if the target function is, for example, `"t^2"` and the algorithm has produced the solution `"t*t"`), the symbolically perfect solution `"t*t"` may still get a (very small) mean square deviation assigned, and the *Bull's-Eye Bonus Factor* never kicks in. Therefore, a (small) *Bull's-Eye Tolerance* value can be defined. If the mean square deviation is equal or smaller than the *Bull's-Eye Bonus Factor*, the solution is considered to have zero mean square deviation and to be a perfect solution.

- Adding the complexity penalty to the mean square deviation can - because of limited numerical precision - "eat away" the last significant digit(s) of the raw fitness value. This means that in the "endgame", when deviations are minuscule, the software could no longer decide which individual is better. Therefore, a mechanism was included in the raw fitness calculation to detect that this has just happened. In such case, the very small mean square deviation is amplified by a factor of 10, or 100, or even 1000 (if necessary), and the complexity penalty is decreased accordingly by a small amount so that in total the raw fitness value still decreases for fitter individuals.

8.2.5 Summary

Figure 20 shows the source-code of the new top-level main loop. In line 29, the new parameters `BULLS_EYE_BONUS` and `BULLS_EYE_TOLERANCE`, as discussed in Chapter 8.2.4, are defined.

In lines 37, 38, and 39, the repetition probabilities `XOVER_REP_PRB`, `TREEMUT_REP_PRB`, and `POINTMUT_REP_PRB` for cross-over, tree-mutation, and point-mutation are defined.

In lines 56 and 57, the phenotype string of any individual containing Gray-coded constants is converted into a string with decimal representation. Conesequently, in line 64, both the original (Gray-coded) phenotype, and also the phenotype in its decimal representation is written to the log-file.

```
#include "InductiveLawDiscoverer.h"
#include "ga.h"
#define HISTORY 1
int main()
{
   int i;
   int j;
   int minMDD;
   int tmpMDD;
   int maxMDD;
   int countMDD;
   int SmallestSubTreeOfVirtualPop;

   SmallestSubTreeOfVirtualPop = 0;
   minMDD = MIN_MDD;     // minimal "max. derivation depth"
   tmpMDD = START_MDD;   // current (initial) "max. derivation depth"
   maxMDD = MAX_MDD;     // maximum "max. derivation depth"
   countMDD = 1;         // if 1 count up, if -1 count down

   LawDiscovererFarm ILDF(GENEPOOLCOUNT, POPSIZE, HISTORY, EXACTUNIF, LANGUAGEDEF,
                          tmpMDD, SmallestSubTreeOfVirtualPop);

   ILDF.PhysicalModel->SetReferenceFormula(SYS_FORMULA);
   ILDF.PhysicalModel->AddParameter(MYPARAM, PARAM_FROM, PARAM_TO, STEP_WIDTH);
   ILDF.ExchProb = EXCH_PROB;
   ILDF.MinStepsBetweenExch = MIN_STEPS_BE;
   ILDF.NoOfExchgs = NO_OF_EXCHG;
   ILDF.PhysicalModel->CacheRefValues();
   ILDF.SetComplexityPenaltyWeight(P_COMPLEX, BULLS_EYE_BONUS, BULLS_EYE_TOLERANCE);
   ILDF.SetMissingParamPenalty(P_PARAM);
   ILDF.SetFitnessScaling(0, MINIMIZE);
   ILDF.SetFitnessAdjustment(One_DividedBy_OnePlusX);
   ILDF.SetSelection(LINEAR_RANK, TSR_MIN);
   ILDF.SetSampling(STOCHASTIC_UNIVERSAL, POPSIZE);
   ILDF.SetMating(RANDOM_PERMUTATION);
   ILDF.SetSelectionHeuristic(PropSelectionHeuristic, SEARCHSPACESIZE);
   ILDF.SetCrossOver(NO_OF_XOVER_POINTS, XOVER_PROB, XOVER_REP_PRB);
   ILDF.SetTreeMutProb(TREEMUT_PROB, TREEMUT_REP_PRB);
   ILDF.SetPointMutProb(POINTMUT_PROB, POINTMUT_REP_PROB);
   ILDF.SetSaveBest(YES);
   ILDF.ComputeFitnessValues(YES);

   string ThisIndividual;
   string ThisIndividualDec;
   string LastIndividual;
   OutputFile.open(FILENAME);
   OutputFile.precision(numeric_limits<double>::digits10 + 1);
   while(true)
   {
      i++;
      j++;
      if(j==10) // include sub-trees only once every 10 steps
         ILDF.SetSmallestSubTreeOfVirtualPop(minMDD);

      ThisIndividual = ILDF.BestIndividual->PhenoTypeString();
      ThisIndividualDec = ILDF.ILDFarm[LDFarm->BestIDLNumber]->
                          PhysicalModel->Parser->ConvertToDecimal(ThisIndividual);
      if (!(ThisIndividual.compare(LastIndividual) == 0))
      {
         // if best individual different than previously best: write to file
         OutputFile << endl << "Generation " << i << ", Farm " << ILDF.BestIDLNumber;
         OutputFile << ", MDD " << tmpMDD;
         OutputFile << ", Fitness " << ILDF.BestIndividual->RawFitness << ": " << endl;
         OutputFile << ThisIndividual.c_str() << endl << ThisIndividualDec.c_str() << endl;
         OutputFile.flush();
         LastIndividual = ThisIndividual;
      }

      if (tmpMDD >= maxMDD)
         countMDD = -1; // maximum "max. deriv depth" reached -> count down
      if (tmpMDD <= minMDD)
         countMDD = 1;  // minimum "max. deriv depth" reached -> count up
      tmpMDD = tmpMDD + countMDD;
      ILDF.SetMaxDerivDepth(tmpMDD);

      if (j == 10)
      {   // de-activate inclusion of sub-trees for the next 10 steps
         ILDF.SetSmallestSubTreeOfVirtualPop(0);
         j = 0;
      };
      ILDF.NextStep();
   };
}
```

Figure 20: Source code of main loop as used in third simulation round.

9 Third Round of GP Simulations

This chapter describes the third round of GP simulations, where the software package in its further improved version (allowing for 10-ary Gray coded numbers, and including "end-game"-improvements, as described in Chapter 8.2), was used. Figure 20 shows the top-level source code used in this third simulation round.

9.1 Accelerated Point-Like Particle in Homogeneous Gravity Field

In this simulation (internal simulation identifier: GPK32AG), the same physical system as in Chapters 5.2 and 7.2 (accelerated point-like particle in homogeneous gravity field) was simulated. Again, no random noise was superimposed on the simulated measurement values.

9.1.1 Simulation Parameters

Table 12 lists all simulation parameters.

9.1.2 Simulation Result

The generated protocol file is listed in Appendix 13.4.4. The solution found in generation 1457, namely $\mathbf{100 + t \cdot (50 - 4.903325 \cdot t)}$, **exactly matches the target function**.

This simulation nicely demonstrates the effectiveness of the software improvements:

In generation 345, a candidate solution with the right structure, but not yet fitting numerical parameters, surfaces in "Farm 215". Because of the bracket-nesting and the chosen complexity penalty of 1, the best fitness value this individual could ever get is 18.0 (not taking into consideration the new *Bull's Eye Bonus*). Later, in generation 619, another candidate surfaces in "Farm 24". The best fitness value that individual could ever get is 15.0 (again, not taking into consideration the new *Bull's Eye Bonus*). Eventually, in generation 863, "Farm 212" gets the lead with an individual having even less complex bracket-nesting. The best fitness value this individual could ever get is 13.0.

All these sub-simulations run in parallel, and eventually, in generation 1267, "Farm 215" has brooded the first perfectly matching solution. Because of the *Bull's-Eye Bonus Factor* of 0.5, it gets an fitness value of $15 \cdot 0.5 = 9$ assigned, and is therefore immediately recognized as the optimal solution and outranks the less-complex approximations. Only after a while (in generation 1320 and 1457), "Farm 24" and "Farm 212" catch up with equivalent solutions that only get better fitness values because of less complex bracket-nesting.

In all cases, 10-ary Gray-coding allowed the numerical constants to be slowly but steadily evolved towards the optimal values by the mutation operators.

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar E, see Appendix 13.3.5
formula	SYS_FORMULA	100+50*t-9.80665/2*t^2
parameter	MYPARAM	t
range	PARAM_FROM, PARAM_TO	-50 ... 50
step width	STEP_WIDTH	1
no. of parallel gene-pools	GENEPOOLCOUNT	220
population size per pool	POPSIZE	2 500
total no. of individuals	GENEPOOLCOUNT*POPSIZE	550 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	31
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	30
maximum "max. deriv. depth"	MAX_MDD	100
initial "max. deriv. depth"	START_MDD	80, ascending
inclusion of sub-trees	lines 52, 53 and 76-80	min size 30 once every 10 steps
complexity penalty	P_COMPLEX	1
“Bull’s-Eye Bonus Factor”	BULLS_EYE_BONUS	0.5
“Bull’s-Eye Tolerance”	BULLS_EYE_TOLERANCE	1E-24
missing parameter penalty	P_PARAM	5 000
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.2
cross-over repetition probability	XOVER_REP_PRB	0.25
subtree mutation probability	TREEMUT_PROB	0.04
tree-mut. repetition probability	TREEMUT_REP_PRB	0.5
point-mutation probability	POINTMUT_PROB	0.02
point-mut. repetition probability	POINTMUT_REP_PRB	0.5
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1+f_i^{std})$

Table 12: Simulation parameters for “ballistic trajectory”-problem, round three.

9.2 Path of a Satellite Around a Central Gravitational Source in 2D Cartesian Coordinates

In this simulation (internal identifier: GPK34H) the system described by formula 18 (see Chapter 7.4) was tackled once more with the improved software version and modified simulation parameters.

9.2.1 Simulation Parameters

Table 13 shows all simulation parameters.

9.2.2 Simulation Result

The generated protocol is listed in Appendix 13.4.9 and shows the evolution process.

Figure 21 shows the best function found after 964 generations. After that, no better solution came up even after 1000 more generations.

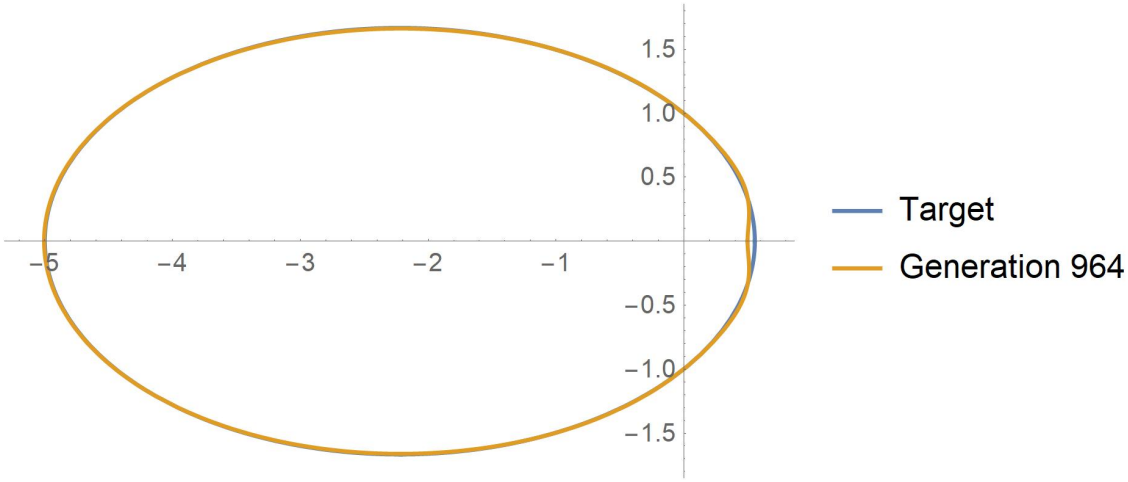

Figure 21: The approximate solution for the "Elliptic Satellite Path in 2D Cartesian Coordinates"-problem found in round three.

This is the approximate solution as found in generation 964, as as depicted in Figure 21:

$$\vec{r}(\varphi) = \begin{pmatrix} \frac{\arccos(\sin(\sin(\cos(\varphi))))}{\cos(\cos(\cos(\cos(\sin(\varphi)))))} \cos(\varphi) \\ \frac{\arccos(\sin(\sin(\cos(\varphi))))}{\cos(\cos(\cos(\cos(\sin(\varphi)))))} \sin(\varphi) \end{pmatrix} \qquad (20)$$

As can be seen by comparing Figure 21 and Figure 19, this approximation fits the target function better than the one previously generated (see Chapter 7.4).

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar G, see Appendix 13.3.7
formula	SYS_FORMULA	Cos(phi)/(1+0.8*Cos(phi)): Sin(phi)/(1+0.8*Cos(phi))
parameter	MYPARAM	phi
range	PARAM_FROM, PARAM_TO	$0 \ldots 4 \cdot 3.14159$
step width	STEP_WIDTH	3.14159/12
no. of parallel gene-pools	GENEPOOLCOUNT	180
population size per pool	POPSIZE	2 000
total no. of individuals	GENEPOOLCOUNT*POPSIZE	360 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	31
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	50
maximum "max. deriv. depth"	MAX_MDD	160
initial "max. deriv. depth"	START_MDD	120, ascending
inclusion of sub-trees	lines 52, 53 and 76-80	min size 30 once every 10 steps
complexity penalty	P_COMPLEX	0.0001
“Bull’s-Eye Bonus Factor”	BULLS_EYE_BONUS	0.5
“Bull’s-Eye Tolerance”	BULLS_EYE_TOLERANCE	1E-24
missing parameter penalty	P_PARAM	5 000
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.2
cross-over repetition probability	XOVER_REP_PRB	0.25
subtree mutation probability	TREEMUT_PROB	0.04
tree-mut. repetition probability	TREEMUT_REP_PRB	0.5
point-mutation probability	POINTMUT_PROB	0.03
point-mut. repetition probability	POINTMUT_REP_PRB	0.5
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1 + f_i^{std})$

Table 13: Simulation parameters for “elliptic satellite path in 2D Cartesian coordinates”-problem, round three.

10 Round Four: Simulating Noisy Measurement Values

This chapter describes the fourth and final round of GP simulations. The same software version as in the previous chapter was used for these simulations. However, different to the simulations so far, normally distributed random numbers were added to the simulated reference values, thereby simulating measurement errors.

Figure 22 shows the modified main source code. In line 23, Gaussian random noise is added to the values of the reference formula. The parameter `ABS_NOISE_SIGMA` is the σ-value of the normally distributed (pseudo) random numbers. The normally distributed pseudo random numbers are generated from uniformly distributed pseudo random numbers (as provided by class `Uniran`) by the "Box-Muller transform" method, as described in [Box and Muller, 1958, p.610-611].

In line 48, all reference values (both undistorted values and "noisy" ones) are dumped to the log file.

10.1 Force-Free Point-Like Particle

In this simulation (internal simulation identifier: GPK3N1B), the same physical system as in Chapters 5.1 and 7.1 (one-dimensional motion of a force-free point-like particle) was simulated. In contrast to the above-mentioned simulations, this time normally distributed random numbers with a σ-value of 1 were added to the reference values. Table 14 shows both the undistorted and noisy values.

10.1.1 Simulation Parameters

Table 15 lists all simulation parameters.

10.1.2 Simulation Result

The generated protocol file is listed in Appendix 13.4.10. The solution found in generation 1515 is:

$$122.87252998 + 4.59646859 \cdot t \tag{21}$$

Conventional linear regression[2] yields the following result:

$$122.87253047 + 4.5964685002 \cdot t \tag{22}$$

The numeric parameters calculated with GP on one hand and linear regression on the other hand agree in at least 7 significant digits.

[2]Calculated with Mathematica 11.1, command **SetPrecision[FindFit[data, {a*t+b},{a,b},t],11]**

```
#include "InductiveLawDiscoverer.h"
#include "ga.h"
#define HISTORY 1
int main()
{
   int i;
   int j;
   int minMDD;
   int tmpMDD;
   int maxMDD;
   int countMDD;
   int SmallestSubTreeOfVirtualPop;

   SmallestSubTreeOfVirtualPop = 0;
   minMDD = MIN_MDD;     // minimal "max. derivation depth"
   tmpMDD = START_MDD;   // current (initial) "max. derivation depth"
   maxMDD = MAX_MDD;     // maximum "max. derivation depth"
   countMDD = 1;         // if 1 count up, if -1 count down

   LawDiscovererFarm ILDF(GENEPOOLCOUNT, POPSIZE, HISTORY, EXACTUNIF, LANGUAGEDEF,
                          tmpMDD, SmallestSubTreeOfVirtualPop);
   ILDF.PhysicalModel->SetReferenceFormula(SYS_FORMULA);
   ILDF.AddNoiseParameter(ABS_NOISE_SIGMA,true,0.0,true);  // Add noise to reference values
   ILDF.PhysicalModel->AddParameter(MYPARAM, PARAM_FROM, PARAM_TO, STEP_WIDTH);
   ILDF.ExchProb = EXCH_PROB;
   ILDF.MinStepsBetweenExch = MIN_STEPS_BE;
   ILDF.NoOfExchgs = NO_OF_EXCHG;
   ILDF.PhysicalModel->CacheRefValues();
   ILDF.SetComplexityPenaltyWeight(P_COMPLEX, BULLS_EYE_BONUS, BULLS_EYE_TOLERANCE);
   ILDF.SetMissingParamPenalty(P_PARAM);
   ILDF.SetFitnessScaling(0, MINIMIZE);
   ILDF.SetFitnessAdjustment(One_DividedBy_OnePlusX);
   ILDF.SetSelection(LINEAR_RANK, TSR_MIN);
   ILDF.SetSampling(STOCHASTIC_UNIVERSAL, POPSIZE);
   ILDF.SetMating(RANDOM_PERMUTATION);
   ILDF.SetSelectionHeuristic(PropSelectionHeuristic, SEARCHSPACESIZE);
   ILDF.SetCrossOver(NO_OF_XOVER_POINTS, XOVER_PROB, XOVER_REP_PRB);
   ILDF.SetTreeMutProb(TREEMUT_PROB, TREEMUT_REP_PRB);
   ILDF.SetPointMutProb(POINTMUT_PROB, POINTMUT_REP_PROB);
   ILDF.SetSaveBest(YES);
   ILDF.ComputeFitnessValues(YES);

   string ThisIndividual;
   string ThisIndividualDec;
   string LastIndividual;
   OutputFile.open(FILENAME);
   OutputFile.precision(numeric_limits<double>::digits10 + 1);
   OutputFile << ILDF.GetRefValues() << endl;  // write "noisy" reference values to log file
   while(true)
   {
      i++;
      j++;
      if(j==10) // include sub-trees only once every 10 steps
         ILDF.SetSmallestSubTreeOfVirtualPop(minMDD);

      ThisIndividual = ILDF.BestIndividual->PhenoTypeString();
      ThisIndividualDec = ILDF.ILDFarm[LDFarm->BestIDLNumber]->
                          PhysicalModel->Parser->ConvertToDecimal(ThisIndividual);
      if (!(ThisIndividual.compare(LastIndividual) == 0))
      {
         // if best individual different than previously best: write to file
         OutputFile << endl << "Generation " << i << ", Farm " << ILDF.BestIDLNumber;
         OutputFile << ", MDD " << tmpMDD;
         OutputFile << ", Fitness " << ILDF.BestIndividual->RawFitness << ": " << endl;
         OutputFile << ThisIndividual.c_str() << endl << ThisIndividualDec.c_str() << endl;
         OutputFile.flush();
         LastIndividual = ThisIndividual;
      }

      if (tmpMDD >= maxMDD)
         countMDD = -1; // maximum "max. deriv depth" reached -> count down
      if (tmpMDD <= minMDD)
         countMDD = 1;  // minimum "max. deriv depth" reached -> count up
      tmpMDD = tmpMDD + countMDD;
      ILDF.SetMaxDerivDepth(tmpMDD);
      if (j == 10)
      {   // de-activate inclusion of sub-trees for the next 10 steps
         ILDF.SetSmallestSubTreeOfVirtualPop(0);
         j = 0;
      };
      ILDF.NextStep();
   };
}
```

Figure 22: Source code of main loop as used in fourth simulation round.

t	actual value	noisy value	t	actual value	noisy value	t	actual value	noisy value
0.00	123.000	123.2546948	3.35	138.343	138.4856319	6.70	153.686	154.9225389
0.05	123.229	121.7722918	3.40	138.572	138.3885421	6.75	153.915	153.4114514
0.10	123.458	123.6161667	3.45	138.801	138.0317369	6.80	154.144	152.7457015
0.15	123.687	122.3453976	3.50	139.030	140.3427389	6.85	154.373	156.0519464
0.20	123.916	124.5769908	3.55	139.259	139.5936280	6.90	154.602	154.0865304
0.25	124.145	124.4165600	3.60	139.488	139.0238705	6.95	154.831	154.7316593
0.30	124.374	124.2521042	3.65	139.717	138.8427353	7.00	155.060	154.1767222
0.35	124.603	124.8498198	3.70	139.946	138.7528083	7.05	155.289	155.0065074
0.40	124.832	124.4360401	3.75	140.175	141.4603587	7.10	155.518	155.0172291
0.45	125.061	125.3509735	3.80	140.404	138.6782017	7.15	155.747	156.3895717
0.50	125.290	125.5827896	3.85	140.633	139.0901794	7.20	155.976	155.6234955
0.55	125.519	126.5915317	3.90	140.862	143.3678338	7.25	156.205	156.9555938
0.60	125.748	125.8742282	3.95	141.091	140.2511043	7.30	156.434	155.9013928
0.65	125.977	125.5251027	4.00	141.320	141.6718148	7.35	156.663	157.5239879
0.70	126.206	125.5049455	4.05	141.549	142.0468309	7.40	156.892	156.9669882
0.75	126.435	124.5831433	4.10	141.778	141.7833298	7.45	157.121	157.7761283
0.80	126.664	126.4371712	4.15	142.007	141.4244530	7.50	157.350	156.3099118
0.85	126.893	124.9211466	4.20	142.236	140.5959623	7.55	157.579	156.3542456
0.90	127.122	128.2308552	4.25	142.465	143.7158432	7.60	157.808	158.4618213
0.95	127.351	125.0476098	4.30	142.694	142.9890696	7.65	158.037	158.1866006
1.00	127.580	126.4255114	4.35	142.923	140.9296625	7.70	158.266	156.7139212
1.05	127.809	127.2369790	4.40	143.152	143.6552936	7.75	158.495	160.9655232
1.10	128.038	127.6850312	4.45	143.381	141.2102372	7.80	158.724	159.3247347
1.15	128.267	128.1426853	4.50	143.610	143.8749785	7.85	158.953	160.2179203
1.20	128.496	128.5363886	4.55	143.839	143.0732781	7.90	159.182	158.4881647
1.25	128.725	129.3733873	4.60	144.068	143.5537400	7.95	159.411	158.9442034
1.30	128.954	129.6641025	4.65	144.297	146.5651956	8.00	159.640	160.2211884
1.35	129.183	129.1992849	4.70	144.526	144.4896847	8.05	159.869	157.5145307
1.40	129.412	131.5549363	4.75	144.755	146.9867805	8.10	160.098	159.3988123
1.45	129.641	127.8034657	4.80	144.984	143.8248605	8.15	160.327	157.9161286
1.50	129.870	128.7318791	4.85	145.213	146.3636957	8.20	160.556	161.9631407
1.55	130.099	128.9314944	4.90	145.442	144.8137455	8.25	160.785	160.4497318
1.60	130.328	129.5505455	4.95	145.671	146.0781647	8.30	161.014	160.2184258
1.65	130.557	130.5073188	5.00	145.900	144.0763516	8.35	161.243	163.2165983
1.70	130.786	129.5505100	5.05	146.129	145.8706300	8.40	161.472	160.4847468
1.75	131.015	131.3140028	5.10	146.358	146.4176632	8.45	161.701	161.5695382
1.80	131.244	131.1867984	5.15	146.587	147.7136947	8.50	161.930	161.2106729
1.85	131.473	132.0471614	5.20	146.816	148.1858847	8.55	162.159	161.2486826
1.90	131.702	130.9190481	5.25	147.045	147.9635883	8.60	162.388	160.8372211
1.95	131.931	130.6371494	5.30	147.274	148.6528457	8.65	162.617	164.7736771
2.00	132.160	133.0239644	5.35	147.503	148.9915410	8.70	162.846	162.8607842
2.05	132.389	131.5289953	5.40	147.732	146.6762040	8.75	163.075	162.6202302
2.10	132.618	133.0341510	5.45	147.961	150.2927159	8.80	163.304	162.5147046
2.15	132.847	132.8672766	5.50	148.190	148.3276586	8.85	163.533	162.6765339
2.20	133.076	133.2888561	5.55	148.419	150.0143249	8.90	163.762	165.0440523
2.25	133.305	133.3689039	5.60	148.648	149.2783717	8.95	163.991	165.1979750
2.30	133.534	134.5138767	5.65	148.877	148.7332458	9.00	164.220	163.0470333
2.35	133.763	134.3441211	5.70	149.106	149.4644417	9.05	164.449	163.0952149
2.40	133.992	133.2124554	5.75	149.335	150.6469423	9.10	164.678	164.6490053
2.45	134.221	135.6130820	5.80	149.564	149.1137656	9.15	164.907	163.0328693
2.50	134.450	133.7584939	5.85	149.793	150.3632073	9.20	165.136	163.4773162
2.55	134.679	134.3959257	5.90	150.022	150.3907190	9.25	165.365	165.4656708
2.60	134.908	132.9505243	5.95	150.251	150.5984118	9.30	165.594	165.2460993
2.65	135.137	135.4498679	6.00	150.480	150.4817461	9.35	165.823	164.7875926
2.70	135.366	134.4875896	6.05	150.709	150.7869714	9.40	166.052	165.8792096
2.75	135.595	133.4002557	6.10	150.938	152.0650893	9.45	166.281	168.8575584
2.80	135.824	135.2426169	6.15	151.167	151.5234421	9.50	166.510	166.9155151
2.85	136.053	135.2186518	6.20	151.396	150.9805208	9.55	166.739	166.9382444
2.90	136.282	136.6491903	6.25	151.625	151.3226297	9.60	166.968	166.0861899
2.95	136.511	137.5112998	6.30	151.854	152.4652738	9.65	167.197	166.6652356
3.00	136.740	139.0323000	6.35	152.083	153.5015249	9.70	167.426	167.6724895
3.05	136.969	138.6082211	6.40	152.312	152.2540837	9.75	167.655	166.5830465
3.10	137.198	135.9426428	6.45	152.541	151.4915960	9.80	167.884	167.1160841
3.15	137.427	138.5645659	6.50	152.770	152.1716450	9.85	168.113	169.0532828
3.20	137.656	137.4750854	6.55	152.999	154.6317869	9.90	168.342	169.5124081
3.25	137.885	137.1791819	6.60	153.228	153.4612328	9.95	168.571	167.8292066
3.30	138.114	138.5011848	6.65	153.457	154.6948489			

Table 14: Round four, simulation one: Undistorted and distorted reference values.

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar E, see Appendix 13.3.5
formula	SYS_FORMULA	123+4.58*t
uniform random noise, absolute σ-value	ABS_NOISE_SIGMA	1
parameter	MYPARAM	t
range	PARAM_FROM, PARAM_TO	0 ... 10
step width	STEP_WIDTH	0.05
no. of parallel gene-pools	GENEPOOLCOUNT	70
population size per pool	POPSIZE	5 000
total no. of individuals	GENEPOOLCOUNT*POPSIZE	350 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	17
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	20
maximum "max. deriv. depth"	MAX_MDD	50
initial "max. deriv. depth"	START_MDD	30, ascending
inclusion of sub-trees	lines 53, 54 and 76-80	min size 30 once every 10 steps
complexity penalty	P_COMPLEX	10
“Bull’s-Eye Bonus Factor”	BULLS_EYE_BONUS	0.5
“Bull’s-Eye Tolerance”	BULLS_EYE_TOLERANCE	1E-24
missing parameter penalty	P_PARAM	5 000
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.15
cross-over repetition probability	XOVER_REP_PRB	0.25
subtree mutation probability	TREEMUT_PROB	0.04
tree-mut. repetition probability	TREEMUT_REP_PRB	0.5
point-mutation probability	POINTMUT_PROB	0.02
point-mut. repetition probability	POINTMUT_REP_PRB	0.5
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1+f_i^{std})$

Table 15: Simulation parameters for “force-free point-like particle”-problem, noisy values, round four.

10.2 Accelerated Point-Like Particle in Homogeneous Gravity Field

In this simulation (internal simulation identifier: GPK3N2A), the same physical system as in Chapters 5.2, 7.2 and 9.1 (accelerated point-like particle in homogeneous gravity field) was simulated. Normally distributed random numbers with a σ-value of 0.5 were added to the reference values. Table 16 shows both the undistorted and noisy values.

10.2.1 Simulation Parameters

Table 17 lists all simulation parameters.

10.2.2 Simulation Result

The generated protocol file is listed in Appendix 13.4.11. The solution found in generation 2094 is:

$$100.06519339 - (t \cdot 4.90343062 \cdot (t - 10.19711923)) \tag{23}$$

This is the expanded expression:

$$100.06519339 + 50.000866668 \cdot t - 4.9034306200 \cdot t^2 \tag{24}$$

Conventional regression to a quadratic polynomial[3] yields the following result:

$$100.06552348 + 50.000783632 \cdot t - 4.9034304536 \cdot t^2 \tag{25}$$

The numeric parameters calculated with GP on one hand and regression to a quadratic polynomial on the other hand all agree in at least 5 significant digits.

[3] Calculated with Mathematica 11.1,
command **SetPrecision[FindFit[data, {a + b * t + c * t^2},{a,b,c},t],11]**

t	actual value	noisy value	t	actual value	noisy value	t	actual value	noisy value
-50.0	-14658.312500	-14658.185150	-16.5	-2059.930231	-2059.858915	17.0	-467.060925	-466.442656
-49.5	-14389.372080	-14390.100440	-16.0	-1955.251200	-1955.342929	17.5	-526.643281	-526.895056
-49.0	-14122.883320	-14122.804240	-15.5	-1853.023831	-1853.408463	18.0	-588.677300	-589.376449
-48.5	-13858.846230	-13859.517030	-15.0	-1753.248125	-1752.591756	18.5	-653.162981	-652.323508
-48.0	-13597.260800	-13596.930300	-14.5	-1655.924081	-1655.756767	19.0	-720.100325	-720.358060
-47.5	-13338.127030	-13337.991250	-14.0	-1561.051700	-1561.283765	19.5	-789.489331	-789.539002
-47.0	-13081.444920	-13081.505870	-13.5	-1468.630981	-1469.068114	20.0	-861.330000	-861.771639
-46.5	-12827.214480	-12827.091070	-13.0	-1378.661925	-1379.258521	20.5	-935.622331	-935.763578
-46.0	-12575.435700	-12575.633680	-12.5	-1291.144531	-1290.501852	21.0	-1012.366325	-1012.616710
-45.5	-12326.108580	-12325.963590	-12.0	-1206.078800	-1206.941699	21.5	-1091.561981	-1091.240695
-45.0	-12079.233120	-12079.086730	-11.5	-1123.464731	-1124.236142	22.0	-1173.209300	-1173.385552
-44.5	-11834.809330	-11834.273070	-11.0	-1043.302325	-1042.049408	22.5	-1257.308281	-1256.932984
-44.0	-11592.837200	-11592.774090	-10.5	-965.591581	-966.011529	23.0	-1343.858925	-1344.125229
-43.5	-11353.316730	-11353.542680	-10.0	-890.332500	-890.156593	23.5	-1432.861231	-1432.430737
-43.0	-11116.247920	-11116.598450	-9.5	-817.525081	-817.276166	24.0	-1524.315200	-1524.277706
-42.5	-10881.630780	-10882.556710	-9.0	-747.169325	-747.166660	24.5	-1618.220831	-1617.893267
-42.0	-10649.465300	-10649.578710	-8.5	-679.265231	-679.556505	25.0	-1714.578125	-1715.098169
-41.5	-10419.751480	-10420.737410	-8.0	-613.812800	-614.632819	25.5	-1813.387081	-1813.999458
-41.0	-10192.489320	-10191.934900	-7.5	-550.812031	-550.186610	26.0	-1914.647700	-1914.320789
-40.5	-9967.678831	-9968.830526	-7.0	-490.262925	-490.115390	26.5	-2018.359981	-2018.285181
-40.0	-9745.320000	-9745.897244	-6.5	-432.165481	-433.162150	27.0	-2124.523925	-2125.299964
-39.5	-9525.412831	-9525.698842	-6.0	-376.519700	-376.268053	27.5	-2233.139531	-2231.904270
-39.0	-9307.957325	-9308.133809	-5.5	-323.325581	-324.410963	28.0	-2344.206800	-2343.906433
-38.5	-9092.953481	-9093.015639	-5.0	-272.583125	-272.450636	28.5	-2457.725731	-2457.093271
-38.0	-8880.401300	-8880.381106	-4.5	-224.292331	-224.675192	29.0	-2573.696325	-2574.043243
-37.5	-8670.300781	-8669.976588	-4.0	-178.453200	-178.710330	29.5	-2692.118581	-2692.351980
-37.0	-8462.651925	-8462.296874	-3.5	-135.065731	-133.931634	30.0	-2812.992500	-2812.701906
-36.5	-8257.454731	-8257.446589	-3.0	-94.129925	-94.148083	30.5	-2936.318081	-2937.495316
-36.0	-8054.709200	-8053.637732	-2.5	-55.645781	-54.529891	31.0	-3062.095325	-3062.444919
-35.5	-7854.415331	-7855.334098	-2.0	-19.613300	-20.192870	31.5	-3190.324231	-3191.529667
-35.0	-7656.573125	-7657.142185	-1.5	13.967519	14.542867	32.0	-3321.004800	-3320.301230
-34.5	-7461.182581	-7461.766334	-1.0	45.096675	44.782548	32.5	-3454.137031	-3454.304665
-34.0	-7268.243700	-7268.632427	-0.5	73.774169	73.977751	33.0	-3589.720925	-3590.118712
-33.5	-7077.756481	-7077.781322	0.0	100.000000	99.088176	33.5	-3727.756481	-3726.769682
-33.0	-6889.720925	-6890.338670	0.5	123.774169	123.644984	34.0	-3868.243700	-3868.737327
-32.5	-6704.137031	-6703.987530	1.0	145.096675	145.126507	34.5	-4011.182581	-4011.248312
-32.0	-6521.004800	-6521.033401	1.5	163.967519	164.530866	35.0	-4156.573125	-4156.932789
-31.5	-6340.324231	-6340.037151	2.0	180.386700	181.071642	35.5	-4304.415331	-4304.870490
-31.0	-6162.095325	-6162.486801	2.5	194.354219	194.813513	36.0	-4454.709200	-4455.484589
-30.5	-5986.318081	-5986.965007	3.0	205.870075	206.559498	36.5	-4607.454731	-4606.376393
-30.0	-5812.992500	-5812.560518	3.5	214.934269	215.678539	37.0	-4762.651925	-4762.644533
-29.5	-5642.118581	-5642.548584	4.0	221.546800	221.018902	37.5	-4920.300781	-4920.528166
-29.0	-5473.696325	-5473.488249	4.5	225.707669	226.873527	38.0	-5080.401300	-5080.795948
-28.5	-5307.725731	-5307.715593	5.0	227.416875	227.485704	38.5	-5242.953481	-5243.381714
-28.0	-5144.206800	-5144.100372	5.5	226.674419	227.472081	39.0	-5407.957325	-5407.316299
-27.5	-4983.139531	-4983.107579	6.0	223.480300	223.795486	39.5	-5575.412831	-5574.809344
-27.0	-4824.523925	-4824.033987	6.5	217.834519	217.762642	40.0	-5745.320000	-5745.906483
-26.5	-4668.359981	-4668.069421	7.0	209.737075	209.916296	40.5	-5917.678831	-5918.355724
-26.0	-4514.647700	-4515.037472	7.5	199.187969	199.843940	41.0	-6092.489325	-6092.503822
-25.5	-4363.387081	-4362.691040	8.0	186.187200	185.962083	41.5	-6269.751481	-6270.688547
-25.0	-4214.578125	-4214.923878	8.5	170.734769	171.019872	42.0	-6449.465300	-6450.294642
-24.5	-4068.220831	-4068.362368	9.0	152.830675	153.015035	42.5	-6631.630781	-6631.580446
-24.0	-3924.315200	-3925.293938	9.5	132.474919	132.648625	43.0	-6816.247925	-6816.421875
-23.5	-3782.861231	-3782.704797	10.0	109.667500	109.668373	43.5	-7003.316731	-7003.834435
-23.0	-3643.858925	-3644.298130	10.5	84.408419	84.447404	44.0	-7192.837200	-7192.923595
-22.5	-3507.308281	-3508.405653	11.0	56.697675	57.261220	44.5	-7384.809331	-7383.521052
-22.0	-3373.209300	-3373.499992	11.5	26.535269	26.713490	45.0	-7579.233125	-7579.030367
-21.5	-3241.561981	-3241.979155	12.0	-6.078800	-6.286540	45.5	-7776.108581	-7776.008959
-21.0	-3112.366325	-3112.182730	12.5	-41.144531	-41.295716	46.0	-7975.435700	-7975.876605
-20.5	-2985.622331	-2985.122181	13.0	-78.661925	-78.356288	46.5	-8177.214481	-8177.480363
-20.0	-2861.330000	-2860.183850	13.5	-118.630981	-117.921719	47.0	-8381.444925	-8381.321680
-19.5	-2739.489331	-2738.669721	14.0	-161.051700	-161.080658	47.5	-8588.127031	-8588.663008
-19.0	-2620.100325	-2620.728004	14.5	-205.924081	-206.448783	48.0	-8797.260800	-8797.644758
-18.5	-2503.162981	-2502.594198	15.0	-253.248125	-253.547303	48.5	-9008.846231	-9008.376090
-18.0	-2388.677300	-2388.767757	15.5	-303.023831	-302.207438	49.0	-9222.883325	-9222.298121
-17.5	-2276.643281	-2276.996190	16.0	-355.251200	-355.134584	49.5	-9439.372081	-9439.742978
-17.0	-2167.060925	-2166.867333	16.5	-409.930231	-409.311307	50.0	-9658.312500	-9658.428525

Table 16: Round four, simulation two: Undistorted and distorted reference values.

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar E, see Appendix 13.3.5
formula	SYS_FORMULA	100+50*t-9.80665/2*t^2
uniform random noise, absolute σ-value	ABS_NOISE_SIGMA	0.5
parameter	MYPARAM	t
range	PARAM_FROM, PARAM_TO	-50 ... 50
step width	STEP_WIDTH	0.5
no. of parallel gene-pools	GENEPOOLCOUNT	220
population size per pool	POPSIZE	2 500
total no. of individuals	GENEPOOLCOUNT*POPSIZE	550 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	31
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	30
maximum "max. deriv. depth"	MAX_MDD	100
initial "max. deriv. depth"	START_MDD	80, ascending
inclusion of sub-trees	lines 52, 53 and 76-80	min size 30 once every 10 steps
complexity penalty	P_COMPLEX	1
“Bull’s-Eye Bonus Factor”	BULLS_EYE_BONUS	0.5
“Bull’s-Eye Tolerance”	BULLS_EYE_TOLERANCE	1E-24
missing parameter penalty	P_PARAM	5 000
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.2
cross-over repetition probability	XOVER_REP_PRB	0.25
subtree mutation probability	TREEMUT_PROB	0.04
tree-mut. repetition probability	TREEMUT_REP_PRB	0.5
point-mutation probability	POINTMUT_PROB	0.02
point-mut. repetition probability	POINTMUT_REP_PRB	0.5
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1+f_i^{std})$

Table 17: Simulation parameters for “ballistic trajectory”-problem, noisy values, round four.

10.3 Path of a Satellite Around a Central Gravitational Source in 1D Polar Coordinates

In this simulation (internal simulation identifier: GPK3N3E), the same physical system as in Chapters 5.3, and 9.2 (satellite path around a central gravitational source in 1D polar coordinates) was simulated. Normally distributed random numbers with a σ-value of 0.01 were added to the reference values. Table 18 shows both the undistorted and noisy values.

10.3.1 Simulation Parameters

Interestingly, pre-simulations with Grammar F (producing 10-ary Gray-coded numbers, see Appendix 13.3.6) only led to approximate solutions, not matching the structure of the target function. Therefore, we decided to go back to Grammar C (see Appendix 13.3.3), as this grammar was quite successful in the simulation described in Chapter 7.3. Table 19 lists all simulation parameters.

10.3.2 Simulation Result

The generated protocol file is listed in Appendix 13.4.12. The solution found in generation 191 (after which no bettter solutions came up for another 1000 generations) is:

$$\frac{1.250001}{\cos(\varphi) + 1.250059} \tag{26}$$

Conventional regression[4] to the fractional expression $\frac{a}{\cos(\varphi)+b}$ yields the following result:

$$\frac{1.248892}{\cos(\varphi) + 1.249791} \tag{27}$$

The numeric parameters calculated with GP on one hand and conventional regression on the other hand agree in 2 significant digits.

[4] Calculated with Mathematica 11.1,
command **SetPrecision[FindFit[data, {a/(Cos[phi] + b)},{a,b},phi],7]**

φ	actual value	noisy value	φ	actual value	noisy value	φ	actual value	noisy value
0,00	0,5555555556	0,5581025034	4,20	1,6453014860	1,6289011090	8,40	1,7106618190	1,7007892870
0,05	0,5558643047	0,5412972228	4,25	1,5548955690	1,5674040010	8,45	1,8151503770	1,8138357590
0,10	0,5567918400	0,5583735066	4,30	1,4719721870	1,4749228830	8,50	1,9290477790	1,9218545080
0,15	0,5583420352	0,5449260117	4,35	1,3958823600	1,3759489850	8,55	2,0531090600	2,0440058860
0,20	0,5605213819	0,5671312899	4,40	1,3260248080	1,3310577440	8,60	2,1880690790	2,1725612910
0,25	0,5633390429	0,5660546430	4,45	1,2618468530	1,2401392250	8,65	2,3346000350	2,3561668060
0,30	0,5668069277	0,5655879697	4,50	1,2028434830	1,2054932680	8,70	2,4932514940	2,4933993370
0,35	0,5709397914	0,5734079895	4,55	1,1485552540	1,1408980350	8,75	2,6643685380	2,6598208400
0,40	0,5757553581	0,5717957593	4,60	1,0985655190	1,0934229190	8,80	2,8479838330	2,8400908780
0,45	0,5812744701	0,5841742048	4,65	1,0524973150	1,0751792710	8,85	3,0436807220	3,0351160610
0,50	0,5875212659	0,5904491615	4,70	1,0100101410	1,0096469880	8,90	3,2504275740	3,2632480970
0,55	0,5945233869	0,6052487035	4,75	0,9707967614	0,9931145660	8,95	3,4663895090	3,4784592590
0,60	0,6023122193	0,6035745018	4,80	0,9345801496	0,9229887548	9,00	3,6887330420	3,6770033760
0,65	0,6109231699	0,6064041967	4,85	0,9011106002	0,9126175570	9,05	3,9134522910	3,8999144400
0,70	0,6203959833	0,6133854378	4,90	0,8701630606	0,8638805154	9,10	4,1352607690	4,1349708220
0,75	0,6307751028	0,6122565354	4,95	0,8415346818	0,8456063286	9,15	4,3476063320	4,3288650250
0,80	0,6421100796	0,6398417920	5,00	0,8150425901	0,7968061060	9,20	4,5428712440	4,5262844070
0,85	0,6544560368	0,6347375029	5,05	0,7905218728	0,7879381724	9,25	4,7128049370	4,7138116450
0,90	0,6678741945	0,6789627462	5,10	0,7678237652	0,7684203968	9,30	4,8491963080	4,8457173010
0,95	0,6824324618	0,6593985597	5,15	0,7468140237	0,7580809708	9,35	4,9447264800	4,9343724060
1,00	0,6982061051	0,6866612191	5,20	0,7273714715	0,7410703184	9,40	4,9938683690	4,9921404650
1,05	0,7152785013	0,7095582913	5,25	0,7093866990	0,7185725816	9,45	4,9936469070	5,0194124910
1,10	0,7337419851	0,7302122973	5,30	0,6927609061	0,7065493632	9,50	4,9440757050	4,9481308560
1,15	0,7536988036	0,7524556565	5,35	0,6774048713	0,6922902814	9,55	4,8481550110	4,8501474560
1,20	0,7752621876	0,7756660737	5,40	0,6632380352	0,6526800750	9,60	4,7114315240	4,7026134230
1,25	0,7985575560	0,8050414288	5,45	0,6501876850	0,6735048444	9,65	4,5412360620	4,5359184180
1,30	0,8237238648	0,8308248900	5,50	0,6381882318	0,6395648183	9,70	4,3457836420	4,3482485360
1,35	0,8509151190	0,8510779683	5,55	0,6271805686	0,6431338178	9,75	4,1333218360	4,1226023010
1,40	0,8803020593	0,9017314222	5,60	0,6171115012	0,6234152186	9,80	3,9114604580	3,9037812980
1,45	0,9120740417	0,8936986989	5,65	0,6079332452	0,6064957029	9,85	3,6867409710	3,6961438000
1,50	0,9464411227	0,9350599140	5,70	0,5996029808	0,6031873983	9,90	3,4644383280	3,4761424090
1,55	0,9836363610	0,9719613050	5,75	0,5920824601	0,6052018828	9,95	3,2485473840	3,2411294500
1,60	1,0239183410	1,0161437970	5,80	0,5853376614	0,5808353173	10,00	3,0418919310	3,0395714370
1,65	1,0675739180	1,0670771060	5,85	0,5793384873	0,5850405605	10,05	2,8462988490	2,8420884150
1,70	1,1149211630	1,1025662630	5,90	0,5740584992	0,5777456896	10,10	2,6627935030	2,6790078710
1,75	1,1663124790	1,1693025070	5,95	0,5694746890	0,5729488072	10,15	2,4917878720	2,4896355780
1,80	1,2221378180	1,2215658020	6,00	0,5655672812	0,5655847424	10,20	2,3332459540	2,3382202000
1,85	1,2828278820	1,2885694970	6,05	0,5623195656	0,5630992801	10,25	2,1868204200	2,1901094210
1,90	1,3488571570	1,3410276380	6,10	0,5597177565	0,5709886492	10,30	2,0519602890	2,0360538580
1,95	1,4207464950	1,4078079880	6,15	0,5577508780	0,5613152992	10,35	1,9279925700	1,9260803010
2,00	1,4990648890	1,5077045330	6,20	0,5564106729	0,5522558810	10,40	1,8141820610	1,8409796660
2,05	1,5844298790	1,5758298320	6,25	0,5556915349	0,5526678319	10,45	1,7097736950	1,7171478050
2,10	1,6775058250	1,6816673350	6,30	0,5555904623	0,5617032004	10,50	1,6140214370	1,6135443240
2,15	1,7789989850	1,7792017510	6,35	0,5561070336	0,5702922830	10,55	1,5262071140	1,5059874530
2,20	1,8896479380	1,8917764990	6,40	0,5572434040	0,5566642407	10,60	1,4456519150	1,4496734540
2,25	2,0102074240	2,0108464630	6,45	0,5590043231	0,5485102828	10,65	1,3717226480	1,3941130380
2,30	2,1414230790	2,1512218460	6,50	0,5613971737	0,5554136232	10,70	1,3038343850	1,2925734700
2,35	2,2839939110	2,2898051220	6,55	0,5644320327	0,5807599016	10,75	1,2414506540	1,2483192700
2,40	2,4385186800	2,4307232340	6,60	0,5681217543	0,5704540825	10,80	1,1840820420	1,1963862790
2,45	2,6054219050	2,6193427250	6,65	0,5724820771	0,5848605657	10,85	1,1312838290	1,1378481390
2,50	2,7848551190	2,7779400580	6,70	0,5775317559	0,5898971452	10,90	1,0826530710	1,0762472370
2,55	2,9765698840	2,9737391410	6,75	0,5832927201	0,5782572344	10,95	1,0378254260	1,0523276420
2,60	3,1797614820	3,1601867250	6,80	0,5897902615	0,5758072763	11,00	0,9964719329	0,9977213604
2,65	3,3928870590	3,3960157380	6,85	0,5970532523	0,6138427158	11,05	0,9582958487	0,9465117123
2,70	3,6134701820	3,6046860780	6,90	0,6051143978	0,5999597022	11,10	0,9230296456	0,9025245499
2,75	3,8379156550	3,8159682130	6,95	0,6140105264	0,6130171193	11,15	0,8904321963	0,8849898053
2,80	4,0613734040	4,0555595730	7,00	0,6237829202	0,6149501420	11,20	0,8602861774	0,8718258400
2,85	4,2777050170	4,2693615350	7,05	0,6344776919	0,6316527658	11,25	0,8323956950	0,8209534372
2,90	4,4796150390	4,4832869420	7,10	0,6461462118	0,6411385030	11,30	0,8065841304	0,8217921006
2,95	4,6590021010	4,6690050990	7,15	0,6588455917	0,6652713087	11,35	0,7826921939	0,7716093895
3,00	4,8075535650	4,8304765650	7,20	0,6726392315	0,6691141869	11,40	0,7605761770	0,7711401609
3,05	4,9175489230	4,9339411350	7,25	0,6875974358	0,6951033735	11,45	0,7401063846	0,7124994658
3,10	4,9827626360	4,9702090650	7,30	0,7037981100	0,6984720381	11,50	0,7211657348	0,7165787616
3,15	4,9992932690	5,0106689290	7,35	0,7213275439	0,7299374226	11,55	0,7036485086	0,7126103080
3,20	4,9661265630	4,9643174170	7,40	0,7402812936	0,7410311755	11,60	0,6874592362	0,6963808753
3,25	4,8852871280	4,8782289470	7,45	0,7607651740	0,7673164571	11,65	0,6725117061	0,6705693739
3,30	4,7615378040	4,7654096530	7,50	0,7828963734	0,7724954914	11,70	0,6587280838	0,6481932583
3,35	4,6017071280	4,6031334480	7,55	0,8068047048	0,7945571606	11,75	0,6460381288	0,6315550439
3,40	4,4138136380	4,4119790600	7,60	0,8326340081	0,8391722214	11,80	0,6343784998	0,6265145487
3,45	4,2061797240	4,1984870930	7,65	0,8605437188	0,8620397250	11,85	0,6236921383	0,6204664163
3,50	3,9866900340	3,9998174230	7,70	0,8907106180	0,8751898304	11,90	0,6139277226	0,6125488412
3,55	3,7622782620	3,7656245420	7,75	0,9233307801	0,9480360122	11,95	0,6050391848	0,5934818714
3,60	3,5386547280	3,5340134330	7,80	0,9586217292	0,9646290760	12,00	0,5969852839	0,5840877229
3,65	3,3202374110	3,3114947640	7,85	0,9968248151	1,0094740180	12,05	0,5897292296	0,6136802917
3,70	3,1102266700	3,0982947530	7,90	1,0382078130	1,0312694600	12,10	0,5832383518	0,5845843209
3,75	2,9107631550	2,9236167420	7,95	1,0830677370	1,0783997720	12,15	0,5774838111	0,5731576461
3,80	2,7231199910	2,7058620070	8,00	1,1317338510	1,1375457350	12,20	0,5724403466	0,5674952801
3,85	2,5478956440	2,5324674380	8,05	1,1845708230	1,1610261300	12,25	0,5680860577	0,5671910400
3,90	2,3851880400	2,4102463780	8,10	1,2419819550	1,2349900780	12,30	0,5644022172	0,5618080768
3,95	2,2347411290	2,2263421720	8,15	1,3044123530	1,2803036400	12,35	0,5613731126	0,5363379076
4,00	2,0960620930	2,0995802400	8,20	1,3723518370	1,3864232440	12,40	0,5589859149	0,5530001100
4,05	1,9685113040	1,9734896140	8,25	1,4463373060	1,4429846250	12,45	0,5572305714	0,5704528027
4,10	1,8513689780	1,8514222760	8,30	1,5269541210	1,5189983790	12,50	0,5560997228	0,5510394699
4,15	1,7438829090	1,7380574400	8,35	1,6148358930	1,6345718750	12,55	0,5555886428	0,5594829001

Table 18: Round four, simulation three: Undistorted and distorted reference values.

Simulation Parameter	Source-Code	Value
grammar	LANGUAGEDEF	Grammar C, see Appendix 13.3.3
formula	SYS_FORMULA	1/(1+0.8*Cos(phi))
uniform random noise, absolute σ-value	ABS_NOISE_SIGMA	0.01
parameter	MYPARAM	t
range	PARAM_FROM, PARAM_TO	0 ... 4 * 3.14152
step width	STEP_WIDTH	0.05
no. of parallel gene-pools	GENEPOOLCOUNT	50
population size per pool	POPSIZE	5000
total no. of individuals	GENEPOOLCOUNT*POPSIZE	250 000
individual exchange probability	EXCH_PROB	0.2
min. steps between individual exchg.	MIN_STEPS_BE	17
no. of exchanged individuals	NO_OF_EXCHG	1
minimum "max. deriv. depth"	MIN_MDD	20
maximum "max. deriv. depth"	MAX_MDD	60
initial "max. deriv. depth"	START_MDD	40, ascending
inclusion of sub-trees	lines 52, 53 and 76-80	min size 30 once every 10 steps
complexity penalty	P_COMPLEX	1
“Bull’s-Eye Bonus Factor”	BULLS_EYE_BONUS	0
“Bull’s-Eye Tolerance”	BULLS_EYE_TOLERANCE	0
missing parameter penalty	P_PARAM	0.005
selection	LINEAR_RANK	linear rank selection
minimum "target selection rate"	TSR_MIN	0.5
number of cross-over-points	NO_OF_XOVER_POINTS	1
cross-over probability	CROSSOVER_PROB	0.15
cross-over repetition probability	XOVER_REP_PRB	0
subtree mutation probability	TREEMUT_PROB	0.04
tree-mut. repetition probability	TREEMUT_REP_PRB	0
point-mutation probability	POINTMUT_PROB	0.02
point-mut. repetition probability	POINTMUT_REP_PRB	0
elitism	YES	yes
fitness scaling	line 31	only over current population
fitness adjustment	One_DividedBy_OnePlusX	$1/(1+f_i^{std})$

Table 19: Simulation parameters for “elliptic satellite path in 1D polar coordinates”-problem, noisy values, round four.

11 Conclusions

Using software entirely developed by the author, we have demonstrated that an Artificial Intelligence computer program in the domain of Genetic Programming can autonomously derive simple physical theories from (simulated) measurement data sets on a state-of-the-art PC. In three simulations, after some software and simulation parameter improvements, the machine learning algorithm was able to derive each of the exact laws. In a fourth, more complex simulation, an approximate expression for a two-dimensional kinetic problem was found.

The software was also able to handle measurement data series that have been distorted by a normally distributed (Gaussian) random noise overlay, and satisfactorily produced solutions matching the expression calculated by means of ordinary regression methods.

We can confirm that parametrization of GP simulations is a subtle challenge. It depends much on the problem on hand, and is in itself an optimization task. As automated GP parameter optimization requires computing power beyond that of a PC (see, for example, [Fernando et al., 2007]), we had to find working parameters manually, with hints from literature (for example [Poli et al., 2008]), trial-and-error, and ever-growing experience. This can be seen as a disadvantage of the method, as it does not always just work by "one mouse-click".

In conclusion, it should be mentioned that this work has focused on the automated generation of physical laws that can be described in analytic expressions. One must be aware that there are many relevant physical laws that can either be not expressed analytically, or at least where the analytic expression only describes a special case and does not give as much insight as the corresponding differential equation. Therefore, it makes sense to enhance the presented method towards automated generation of differential equations, as, for example, [Hongqing et al., 2000] have already demonstrated.

12 References

[Angeline et al, 1999] P. J. Angeline, et. al., editors. Proceedings of the Congress on Evolutionary Computation, volume 3. Mayflower Hotel, Washington, 1999.

[Baker, 1987] James E. Baker. Reducing bias and inefficiency in the selection algorithms. In: [Grefenstette 1985, p. 101-111]

[Box and Muller, 1958] G. E. P. Box and Mervin E. Muller. A Note on the Generation of Random Normal Deviates. In: *The Annals of Mathematical Statistics (1958), Vol. 29, No. 2*, p. 610–611.

[Caruana et al., 1988] R. A. Caruana, and J .D. Schaffer. Representation and hidden bias: Gray vs. binary coding for genetic algorithms. In [Laird, 1988, p. 183-187]

[Cramer, 1985] Nichael Lynn Cramer. A Representation for the Adaptive Generation of Simple Sequential Programs, in: *Proceedings of the 1st International Conference on Genetic Algorithms, Pages 183-187*. L. Erlbaum Associates Inc. Hillsdale, NJ, 1985.

[Dürrennmatt, 1962] Friedrich Dürrenmatt. Die Physiker. Eine Kommödie in zwei Akten. Peter Schifferli Verlags AG, Zürich, 1962.

[Fernando et al., 2007] Achela K. Fernando and Amithirigala W. Jayawardena. Use of a supercomputer to advance parameter optimisation using genetic algorithms. In: *Journal of Hydroinformatics, 9(4)*, p. 319-329

[Fogel et al, 1966] Lawrence J. Fogel, Alvin J. Owens, Michael John Walsh. Artificial Intelligence through Simulated Evolution. John Wiley & Sons, 1966.

[Geyer-Schulz, 1994] Andreas Geyer-Schulz. Fuzzy Rule-Based Expert Systems and Genetic Machine Learning. Habilitation thesis, Vienna University of Economics and Business Administration. Vienna, 1994.

[Goldberg, 1989] David E. Goldberg. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Weseley, Reading, Massachusetts, 1989

[Gray, 1953] Pulse code communication. U.S. patent 2632058, 1953.

[Grefenstette 1985] John F. Grefenstette (editor). Proceedings of the First International Conference on Genetic Algorithms and their Applications, Lawrence Erlbaum Associates, Hillsdale, New Jersy, 1985.

[Grefenstette et al., 1989] John J. Grefenstette and James E. Baker. How genetic algorithms work: A critical look at implicit parallelism. In [Schaffer, 1989, p. 112-120].

[Holland, 1975] John H. Holland. Adaption in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, Michigan 1975.

[Hongqing et al., 2000] Cao Hongqing, Kang Lishan, Chen Yuping, and Yu Jingxian. Evolutionary Modeling of Systems of Ordinary Differential Equations with Genetic Programming. In: *Genetic Programming and Evolvable Machines, 1, 2000*, p. 309–337, Kluwer Academic Publishers, The Netherlands.

[Hörner, 1996] Helmut Hörner. Ein Kern für genetisches Programmieren in C++. Diploma thesis, Vienna University of Economics and Business Administration. Vienna, 1996.

[Knuth, 1964] Donald E. Knuth. Backus Normal Form vs. Backus Naur Form. *Communications of the ACM*, Volume 7, Issue 12, Pages 735-736. ACM, New York, 1964.

[Koza, 1992] John R. Koza. Genetic Programming: On the Programming of Computers by Means of Natural Selection. The MIT Presss Cambridge, Massachusetts, 1992.

[Koza et al., 1997] John R. Koza, et al. editors. Genetic Programming 1997: Proceedings of the Second Annual Conference. Stanford University, Ca, USA, 1997.

[Laird, 1988] John Laird, editor. Proceedings of the fifth international conference on machine learning. University of Michigan, Ann Arbor, Morgan Kaufmann Publishers, San Mateo, CA, 1988.

[Law and Kelton, 2000] Averill M. Law, W. David Kelton. Simulation Modeling and Analysis. McGraw-Hill, New York, 2000.

[Luke and Spector, 1997] S. Luke and L. Spector. A comparison of crossover and mutation in genetic programming. In [Koza et al., 1997, p. 240-248]

[Mambou et al., 2016] Elie N. Mambou and Theo G. Stewart. Encoding and decoding of Balanced q-ary Sequences Using a Gray Code Prefix. In: *IEEE International Symposium on Information Theory*, pages 380-384, IEEE, Spain, 2016

[Poli et al., 2008] Riccardo Poli, William B. Langdon, Nicholas F. McPhee. A Field Guide to Genetic Programming. With Contributions by John R. Koza. Lightning Source UK, Milton Keynes, 2008.

[Rechenberg, 1973] Ingo Rechenberg. Evolutionsstrategie. *problemata*, Volume 15. Friedrich Frommann Verlag, Stuttgart, 1973.

[Schaffer, 1989] J. David Schaffer, editor. Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo, California, 1989.

[Schöneburg et al., 1994] Eberhard Schöneburg, Frank Heinzmann und Sven Feddersen. Genetische Algorithmen und Evolutionsstrategien: Eine Einführung in die Theorie und Praxis der simulierten Evolution. Addison Weseley, Bonn; Paris; Reading, Massachusetts, 1994.

[Spector et al., 1998] Lee Spector, H. Barnum, H. J. Bernstein, and N. Swamy. Finding a better-than-classical quantum AND/OR algorithm using genetic programming. In [Angeline et al, 1999, pages 2239-2246].

[Spector, 2007] Lee Spector. Automatic Quantum Computer Programming. A Genetic Programming Approach. Springer, 2007.

[Svozil and Svozil, 2016] Alexander Svozil and Karl Svozil., "Induction and physical theory formation by Machine Learning", arXiv: 1609.03862v1, 2016.

13 Appendix

13.1 List of Tables

1 Number of individuals that can be created from the sample grammar in Figure 1 in up to 50 derivation steps. 14
2 Results of various round-one pre-simulations for the “force-free point-like particle”-problem. 34
3 Simulation parameters for “force-free point-like particle”-problem, round one. 35
4 Simulation parameters for the “ballistic trajectory”-problem, round one. . 38
5 Simulation parameters for the “elliptic satellite path in 1D polar coordinates”-problem, round one. 40
6 Evolving expressions as shown in Figure 14. 41
7 Simulation parameters for “force-free point-like particle”-problem, round two. 47
8 Simulation parameters for “accelerated point-like particle in homogeneous gravity field”-problem, round two. 48
9 Simulation parameters for “elliptic satellite path in 1D polar coordinates”-problem, round two. 50
10 Simulation parameters for “elliptic satellite path in 2D Cartesian coordinates”-problem, round two. 52
11 Evolving expressions as shown in Figure 18. 53
12 Simulation parameters for “ballistic trajectory”-problem, round three. . . 61
13 Simulation parameters for “elliptic satellite path in 2D Cartesian coordinates”-problem, round three. 63
14 Round four, simulation one: Undistorted and distorted reference values. . 66
15 Simulation parameters for “force-free point-like particle”-problem, noisy values, round four. 67
16 Round four, simulation two: Undistorted and distorted reference values. . 69
17 Simulation parameters for “ballistic trajectory”-problem, noisy values, round four. 70
18 Round four, simulation three: Undistorted and distorted reference values. 72
19 Simulation parameters for “elliptic satellite path in 1D polar coordinates”-problem, noisy values, round four. 73

13.2 List of Figures

1 A simple grammar. 12
2 A GP “individual”, represented by a simple derivation tree. 13
3 “Roulette wheel” with 3 compartments, each sized according to the respective target sampling rate, from: [Hörner, 1996, p. 34]. 18
4 Stochastic Universal Sampling of 5 individuals, from: [Hörner, 1996, p. 34]. 19
5 UML diagram of the initital software version. 22
6 Sample source code, demonstrating how to use the “naked” GPK. 24
7 Sample source code. 31
8 Top level source code used in simulation round one. 33
9 Round one - evolving expressions for problem “force-free point-like particle”. 36
10 The approximate solution for the “force-free point-like particle”-problem found in round one 36
11 Round one - piece-wise approximate solution for the “ballistic trajectory”-problem. 37
12 Round one - evolving expressions for the “ballistic trajectory”-problem. . 39
13 The approximate solution for the “ballistic trajectory”-problem found in round one. 39
14 Round one - evolving expressions for the “Elliptic Satellite Path in 1D Polar Coordinates”-problem. 41
15 The approximate solution for the “Elliptic Satellite Path in 1D Polar Coordinates”-problem found in round one. 42
16 UML Diagram of improved software package, now including a new top-level class (highlighted in orange) allowing multi-threaded parallel sub-simulations. 44
17 Source code of main loop as used in second simulation round. 45
18 Round two - evolving expressions for the “elliptic satellite path in 2D Cartesian coordinates”-problem. 53
19 The approximate solution for the “Elliptic Satellite Path in 2D Cartesian Coordinates”-problem found in round two. 54
20 Source code of main loop as used in third simulation round. 59
21 The approximate solution for the “Elliptic Satellite Path in 2D Cartesian Coordinates”-problem found in round three. 62
22 Source code of main loop as used in fourth simulation round. 65

13.3 Grammars

13.3.1 Grammar A

```
S := <expression>;

<expression> := <number_pos> | <parameter> | <function0p> |
                <function1p> "(" <expression> ")" |
                <function2p> "(" <expression> "," <expression> ")" |
                "(" <expression> ")" <operator> "(" <expression> ")" |
                "(-" <expression> ")"  ;

<function0p> := "Pi()" ;

<function1p> := "Sin" | "Cos" | "Tan" | "Abs" | "Exp" | "Log" | "ArcSin" | "ArcCos" | "ArcTan" | "Sinh" | "Cosh" | "Tanh" |
                "ArcSinh" | "ArcCosh" | "ArcTanh" | "Sqrt" ;

<function2p> := "Max" | "Min";

<operator> := "+" | "-" | "*" | "/" | "^" ;

<1to9>       := "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;
<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;

<number_pos> := <integer> | <float> | <float_sci>;

<integer> := <digit> |
             <1to9> <digit> |
             <1to9> <digit> <digit> |
             <1to9> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit>;

<float> :=   <digit> "." <1to9> |
             <1to9> <digit> "." <1to9> |
             <1to9> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <digit> "." <digit> <1to9> |
             <1to9> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9>;

<float_sci> := <1to9> <exponent> |
               <1to9> "." <1to9> <exponent> |
               <1to9> "." <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9> <exponent>;

<exponent>  := "E" <sign> "0" <1to9> | "E" <sign> <1to9> <digit> ;
<sign>      := "+" | "-" ;

<parameter> := "t" ;
```

13.3.2 Grammar B

```
S := <expression>;

<expression> := <number_pos> | <parameter> | <function0p> |
                <function1p> "(" <expression> ")" |
                "(" <expression> ")" <operator> "(" <expression> ")" |
                "(-" <expression> ")"  ;

<function0p> := "Pi()" ;

<function1p> := "Sin" | "Cos" | "Tan" | "Abs" | "Exp" | "Log" | "ArcSin" | "ArcCos" | "ArcTan" | "Sinh" | "Cosh" | "Tanh" |
                "ArcSinh" | "ArcCosh" | "ArcTanh" | "Sqrt" ;

<operator> := "+" | "-" | "*" | "/" | "^" ;

<1to9>       := "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;
<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;

<number_pos> := <integer> | <float> | <float_sci>;

<integer> := <digit> |
             <1to9> <digit> |
             <1to9> <digit> <digit> |
             <1to9> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit>;

<float> :=   <digit> "." <1to9> |
             <1to9> <digit> "." <1to9> |
             <1to9> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <digit> "." <digit> <1to9> |
             <1to9> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9>;

<float_sci> := <1to9> <exponent> |
               <1to9> "." <1to9> <exponent> |
               <1to9> "." <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9> <exponent>;

<exponent>  := "E" <sign> "0" <1to9> | "E" <sign> <1to9> <digit> ;
<sign>      := "+" | "-" ;

<parameter> := "t" ;
```

13.3.3 Grammar C

```
S := <expression>;

<expression> := <number_pos> | <parameter> |
                <function1p> "(" <expression> ")" |
                <function2p> "(" <expression> "," <expression> ")" |
                "(" <expression> ")" <operator> "(" <expression> ")" |
                "(-" <expression> ")"  ;

<function1p> := "Sin" | "Cos" | "Tan" | "Abs" | "Exp" | "Log" | "ArcSin" | "ArcCos" | "ArcTan" | "Sinh" | "Cosh" | "Tanh" |
                "ArcSinh" | "ArcCosh" | "ArcTanh" | "Sqrt" ;

<function2p> := "Max" | "Min";

<operator> := "+" | "-" | "*" | "/" | "^" ;

<1to9>       := "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;
<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;

<number_pos> := "Pi()" | <integer> | <float> | <float_sci>;

<integer> := <digit> |
             <1to9> <digit> |
             <1to9> <digit> <digit> |
             <1to9> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit>;

<float> :=   <digit> "." <1to9> |
             <1to9> <digit> "." <1to9> |
             <1to9> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <digit> "." <digit> <1to9> |
             <1to9> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9>;

<float_sci> := <1to9> <exponent> |
               <1to9> "." <1to9> <exponent> |
               <1to9> "." <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9> <exponent>;

<exponent>  := "E" <sign> "0" <1to9> | "E" <sign> <1to9> <digit> ;
<sign>      := "+" | "-" ;

<parameter> := "phi" ;
```

13.3.4 Grammar D

```
S := <vector>;

<vector> := <expression> ":" <expression> ;

<expression> := <number_pos> | <parameter> |
                <function1p> "(" <expression> ")" |
                <function2p> "(" <expression> "," <expression> ")" |
                "(" <expression> ")" <operator> "(" <expression> ")" |
                "(-" <expression> ")"  ;

<function1p> := "Sin" | "Cos" | "Tan" | "Abs" | "Exp" | "Log" | "ArcSin" | "ArcCos" | "ArcTan" | "Sinh" | "Cosh" | "Tanh" |
                "ArcSinh" | "ArcCosh" | "ArcTanh" | "Sqrt" ;

<function2p> := "Max" | "Min";

<operator> := "+" | "-" | "*" | "/" | "^" ;

<1to9>       := "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;
<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;

<number_pos> := "Pi()" | <integer> | <float> | <float_sci>;

<integer> := <digit> |
             <1to9> <digit> |
             <1to9> <digit> <digit> |
             <1to9> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit>;

<float> :=   <digit> "." <1to9> |
             <1to9> <digit> "." <1to9> |
             <1to9> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> <digit> "." <1to9> |
             <digit> "." <digit> <1to9> |
             <1to9> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> <digit> "." <digit> <1to9> |
             <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> <digit> "." <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> <digit> "." <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <1to9> <digit> "." <digit> <digit> <digit> <digit> <digit> <1to9> |
             <digit> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9>;

<float_sci> := <1to9> <exponent> |
               <1to9> "." <1to9> <exponent> |
               <1to9> "." <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <1to9> <exponent> |
               <1to9> "." <digit> <digit> <digit> <digit> <digit> <digit> <1to9> <exponent>;

<exponent>  := "E" <sign> "0" <1to9> | "E" <sign> <1to9> <digit> ;
<sign>      := "+" | "-" ;

<parameter> := "phi" ;
```

13.3.5 Grammar E

```
S := <expression> ;

<expression> := <number_pos> | <parameter> |
                <function1p> "(" <expression> ")" |
                <function2p> "(" <expression> "," <expression> ")" |
                "(" <expression> ")" <operator> "(" <expression> ")" |
                "(-" <expression> ")"  ;

<function1p> := "Sin" | "Cos" | "Tan" | "Abs" | "Exp" | "Log" | "ArcSin" | "ArcCos" | "ArcTan" | "Sinh" | "Cosh" | "Tanh" |
                "ArcSinh" | "ArcCosh" | "ArcTanh" | "Sqrt" ;

<function2p> := "Max" | "Min";

<operator> := "+" | "-" | "*" | "/" | "^" ;

<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;

<number_pos> := "Pi()" |
                "&G" <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                      <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"             |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                    <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"             |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"          |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> "E-08"        |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> "E-08"      |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> "E-08"    |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> "E-08"  |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> "E-08";

<parameter> := "t" ;
```

13.3.6 Grammar F

```
S := <expression> ;

<expression> := <number_pos> | <parameter> |
                <function1p> "(" <expression> ")" |
                <function2p> "(" <expression> "," <expression> ")" |
                "(" <expression> ")" <operator> "(" <expression> ")" |
                "(-" <expression> ")"  ;

<function1p> := "Sin" | "Cos" | "Tan" | "Abs" | "Exp" | "Log" | "ArcSin" | "ArcCos" | "ArcTan" | "Sinh" | "Cosh" | "Tanh" |
                "ArcSinh" | "ArcCosh" | "ArcTanh" | "Sqrt" ;

<function2p> := "Max" | "Min";

<operator> := "+" | "-" | "*" | "/" | "^" ;

<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;

<number_pos> := "Pi()" |
                "&G" <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                      <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"             |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                    <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"             |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"          |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> "E-08"        |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> "E-08"      |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> "E-08"    |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> "E-08"  |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> "E-08";

<parameter> := "phi" ;
```

13.3.7 Grammar G

```
S := <vector>;

<vector> := <expression> ":" <expression> ;

<expression> := <number_pos> | <parameter> |
                <function1p> "(" <expression> ")" |
                <function2p> "(" <expression> "," <expression> ")" |
                "(" <expression> ")" <operator> "(" <expression> ")" |
                "(-" <expression> ")"  ;

<function1p> := "Sin" | "Cos" | "Tan" | "Abs" | "Exp" | "Log" | "ArcSin" | "ArcCos" | "ArcTan" | "Sinh" | "Cosh" | "Tanh" |
                "ArcSinh" | "ArcCosh" | "ArcTanh" | "Sqrt" ;

<function2p> := "Max" | "Min";

<operator> := "+" | "-" | "*" | "/" | "^" ;

<digit>      := "0" | "1" | "2" |"3" |"4" |"5" |"6" |"7" |"8" | "9" ;

<number_pos> := "Pi()" |
                "&G" <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"                |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                      <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"             |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                    <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"             |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> <digit> "E-08"          |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> <digit> "E-08"        |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> <digit> "E-08"      |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> <digit> "E-08"    |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> <digit> "E-08"  |
                "&G" <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0>
                     <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit:0> <digit> "E-08";

<parameter> := "phi" ;
```

13.4 Simulation Protocols

13.4.1 Round 1, Simulation 1

```
Generation 1: Fitness 1642.8: (-((-Log((13.622111)+(t))))+(106.686))
Generation 5: Fitness 1563.33: (-((-Log((53.622111)+(t))))+(106.686))
Generation 7: Fitness 599.891: Max(Sqrt(t),Min(59435,ArcSinh(Abs(5.1362263E+70))))
Generation 9: Fitness 296.357: (t)+(Sqrt(23034.639))
Generation 16: Fitness 262.285: Max(ArcSinh(t),142.36147)
Generation 17: Fitness 178.578: (t)+(Sqrt(20034.639))
Generation 20: Fitness 178.576: (t)+(Sqrt(20034.039))
Generation 27: Fitness 177.774: (t)+(144)
Generation 30: Fitness 172.574: (t)+(143)
Generation 32: Fitness 126.584: (-((-Log((43.622111)^(t))))+(126.686))
Generation 34: Fitness 123.211: (2.3713)*((55.6401)+(t))
Generation 41: Fitness 115.687: (2.5713)*((52.6403)+(t))
Generation 42: Fitness 113.539: (2.5213)*((52.4405)+(t))
Generation 48: Fitness 112.246: (2.5613)*((52.4405)+(t))
Generation 50: Fitness 110.282: (Log((23.322321)^(t)))+(133.286)
Generation 54: Fitness 100.484: (Log((23.322321)^(t)))+(130.286)
Generation 57: Fitness 100.253: (Log((23.522321)^(t)))+(130.286)
Generation 60: Fitness 100.236: (Log((23.522321)^(t)))+(130.226)
Generation 61: Fitness 100.236: (Log((23.522327)^(t)))+(130.226)
Generation 62: Fitness 95.2053: (Log((33.322321)^(t)))+(130.286)
Generation 63: Fitness 95.1855: (Log((33.662109)^(t)))+(130.286)
Generation 64: Fitness 85.4338: (Log((46.662201)^(t)))+(126.651)
Generation 67: Fitness 85.4338: (Log((46.662261)^(t)))+(126.651)
Generation 69: Fitness 85.4337: (Log((46.662201)^(t)))+(126.653)
Generation 70: Fitness 85.4325: (Log((46.666201)^(t)))+(126.651)
Generation 72: Fitness 84.165: (Log((52.6405)^(t)))+(126.686)
Generation 74: Fitness 83.3963: (4.9715)*((25.0401)+(t))
Generation 75: Fitness 83.3534: (4.9713)*((25.0401)+(t))
Generation 77: Fitness 72.8334: (4.9015)*((25.0401)+(t))
Generation 86: Fitness 72.58: (4.9015)*((25.0201)+(t))
Generation 94: Fitness 71.0436: (4.8715)*((25.0401)+(t))
Generation 96: Fitness 70.7549: (4.8513)*((25.0461)+(t))
Generation 103: Fitness 70.6749: (4.7225)*((26.0401)+(t))
Generation 104: Fitness 70.465: (4.7715)*((25.6431)+(t))
Generation 111: Fitness 70.3837: (4.7515)*((25.6431)+(t))
Generation 115: Fitness 70.3273: (4.7513)*((25.7461)+(t))
Generation 116: Fitness 70.1567: (4.7021)*((26.0101)+(t))
Generation 117: Fitness 70.1538: (4.7021)*((26.0141)+(t))
Generation 125: Fitness 70.1497: (4.7021)*((26.0341)+(t))
Generation 131: Fitness 70.1492: (4.7021)*((26.030527)+(t))
Generation 132: Fitness 70.0293: (4.6225)*((26.5401)+(t))
Generation 141: Fitness 70.0208: (4.6255)*((26.5401)+(t))
Generation 145: Fitness 70.0208: (4.6255)*((26.5409)+(t))
Generation 148: Fitness 70.0184: (4.6225)*((26.5671)+(t))
Generation 153: Fitness 70.0182: (4.6225)*((26.5601)+(t))
Generation 155: Fitness 70.0181: (4.6225)*((26.5631)+(t))
Generation 156: Fitness 70.0178: (4.6221)*((26.5671)+(t))
Generation 160: Fitness 70.0172: (4.6215)*((26.5701)+(t))
Generation 161: Fitness 70.0065: (4.6021)*((26.6941)+(t))
Generation 168: Fitness 70.0063: (4.6022)*((26.6941)+(t))
Generation 169: Fitness 70.0059: (4.6021)*((26.6961)+(t))
Generation 171: Fitness 70.0057: (4.6022)*((26.6961)+(t))
Generation 174: Fitness 70.0052: (4.6021)*((26.6991)+(t))
Generation 179: Fitness 70.0051: (4.6021)*((26.7001)+(t))
Generation 186: Fitness 70.0043: (4.6001)*((26.7201)+(t))
Generation 201: Fitness 70.0042: (4.6001)*((26.7141)+(t))
Generation 207: Fitness 70.004: (4.6001)*((26.7161)+(t))
Generation 215: Fitness 70.004: (4.6001)*((26.7171)+(t))
Generation 218: Fitness 70.004: (4.6001)*((26.7164)+(t))
Generation 226: Fitness 70.004: (4.6001)*((26.7169)+(t))
Generation 228: Fitness 70.004: (4.6001)*((26.7166)+(t))
Generation 229: Fitness 70.004: (4.6001)*((26.7167)+(t))
Generation 399: Fitness 70.0024: (4.5701)*((26.9167)+(t))
Generation 404: Fitness 70.0024: (4.5701)*((26.9168)+(t))
Generation 407: Fitness 70.0021: (4.5701)*((26.9177)+(t))
Generation 409: Fitness 70.001: (4.5701)*((26.9267)+(t))
Generation 414: Fitness 70.001: (4.5709)*((26.9167)+(t))
Generation 415: Fitness 70.0006: (4.5721)*((26.9107)+(t))
Generation 429: Fitness 70.0006: (4.5722)*((26.9107)+(t))
Generation 432: Fitness 70.0006: (4.5722)*((26.9102)+(t))
Generation 435: Fitness 70.0006: (4.5723)*((26.9102)+(t))
Generation 450: Fitness 70.0006: (4.5723)*((26.9101)+(t))
Generation 482: Fitness 70.0006: (4.5723)*((26.9091)+(t))
Generation 489: Fitness 70.0006: (4.5733)*((26.9002)+(t))
Generation 498: Fitness 70.0006: (4.5733)*((26.9003)+(t))
Generation 500: Fitness 70.0005: (4.5733)*((26.9012)+(t))
Generation 506: Fitness 70.0005: (4.5731)*((26.9042)+(t))
Generation 507: Fitness 70.0004: (4.5736)*((26.9002)+(t))
Generation 512: Fitness 70.0004: (4.5736)*((26.9003)+(t))
```

```
Generation 513: Fitness 70.0004: (4.5737)*((26.9002)+(t))
Generation 529: Fitness 70.0004: (4.5737)*((26.9001)+(t))
Generation 712: Fitness 70.0004: (4.5737)*((26.9)+(t))
Generation 2366: Fitness 70.0003: (4.5747)*((26.894267)+(t))
Generation 2373: Fitness 70.0003: (4.5747)*((26.894237)+(t))
Generation 2377: Fitness 70.0003: (4.5747)*((26.894217)+(t))
Generation 2380: Fitness 70.0003: (4.5747)*((26.893267)+(t))
Generation 2386: Fitness 70.0003: (4.5747)*((26.893237)+(t))
Generation 2389: Fitness 70.0003: (4.5747)*((26.893167)+(t))
Generation 2393: Fitness 70.0003: (4.5748)*((26.892237)+(t))
Generation 2398: Fitness 70: (4.5787)*((26.864237)+(t))
Generation 2405: Fitness 70: (4.5787)*((26.864837)+(t))
Generation 2416: Fitness 70: (4.5787)*((26.864897)+(t))
Generation 2417: Fitness 70: (4.5788)*((26.864257)+(t))
Generation 2421: Fitness 70: (4.5788)*((26.864237)+(t))
Generation 2425: Fitness 70: (4.5788)*((26.864247)+(t))
Generation 2430: Fitness 70: (4.5788)*((26.864242)+(t))
Generation 2442: Fitness 70: (4.5788)*((26.864244)+(t))
Generation 2743: Fitness 70: (4.5789)*((26.863244)+(t))
Generation 2752: Fitness 70: (4.5789)*((26.863444)+(t))
Generation 2755: Fitness 70: (4.5789)*((26.863644)+(t))
Generation 2757: Fitness 70: (4.5789)*((26.863464)+(t))
Generation 2761: Fitness 70: (4.5789)*((26.863544)+(t))
Generation 2769: Fitness 70: (4.5789)*((26.863548)+(t))
Generation 2851: Fitness 70: (4.5799)*((26.856548)+(t))
Generation 2856: Fitness 70: (4.5799)*((26.856549)+(t))
Generation 2869: Fitness 70: (4.5799)*((26.856559)+(t))
Generation 2870: Fitness 70: (4.5799)*((26.856589)+(t))
Generation 2879: Fitness 70: (4.5799)*((26.856592)+(t))
Generation 2894: Fitness 70: (4.5799)*((26.856591)+(t))
Generation 6997: Fitness 70: (4.5799)*((t)+(26.856591))
Generation 7023: Fitness 70: (4.5799)*((26.856591)+(t))
```

13.4.2 Round 1, Simulation 2

```
Generation 1: Fitness 8605.771805685466: (Cosh((-3.05214)))-(((-301.7075))+(Sinh(5.69)))
Generation 3: Fitness 3657.713907671512: Abs(ArcSinh(((-(7958.5889)-(t)))+(1.792836E+68)))
Generation 8: Fitness 3642.883691217112: Abs(ArcSinh(((-(7958.5889)-(t)))+(5.792836E+68)))
Generation 10: Fitness 3643.190357883779: ArcSinh(((-(7958.5889)-(t)))+(5.792836E+68))
Generation 12: Fitness 3643.323691217112: ArcSinh(((-(7958.5869)-(t)))+(5.792836E+68))
Generation 14: Fitness 3641.86259192995: Abs(ArcSinh(((-(7958.5889)-(t)))+(6.792836E+68)))
Generation 15: Fitness 3637.647893889795: ArcSinh(((-(326.99768)-(t)))+(9.792836E+68))
Generation 16: Fitness 3608.002750145033: ArcSinh(((-(42353)-(t)))+(5.772836E+71))
Generation 17: Fitness 3438.20748102312: (175.5475)-(t)
Generation 20: Fitness 3420.812195654539: ((344.4388)-(171.9823))-(t)
Generation 21: Fitness 3424.32770552312: (173.5475)-(t)
Generation 27: Fitness 3420.601151106456: (172.5475)-(t)
Generation 29: Fitness 3326.851638913397: ((t)*((-Sin((9958.5689)-(t)))))+(163.6692)
Generation 31: Fitness 3328.328433020098: ((t)*((-Sin((-t)))))+(163.6692)
Generation 32: Fitness 3324.398957813856: ((t)*((-Sin((-t)))))+(166.9579)
Generation 36: Fitness 3009.540692125739: ((t)*((-Sqrt(t))))+(169.5509)
Generation 40: Fitness 2740.418718107167: ((t)*((-t)))+(183.6698)
Generation 42: Fitness 2151.555925054603: ((t)*((-t)))+(196.9579)
Generation 50: Fitness 2039.690412192625: ((t)*((-t)))+(201.16826)
Generation 52: Fitness 2039.799104813167: ((t)*((-t)))+(201.16876)
Generation 53: Fitness 2040.825275169833: ((t)*((-t)))+(201.12876)
Generation 56: Fitness 2040.958432580667: ((t)*((-t)))+(201.13876)
Generation 57: Fitness 1999.729914747333: ((t)*((-t)))+(203.13876)
Generation 58: Fitness 1931.479487599888: ((t)*((-t)))+(208.13866)
Generation 59: Fitness 1931.599227367021: ((t)*((-t)))+(208.13869)
Generation 60: Fitness 1924.231203836497: ((t)*((-t)))+(209.12876)
Generation 63: Fitness 1810.595055615844: ((t)*((-t)))+(((t)-((-t)))+(196.9529))
Generation 66: Fitness 1799.040496733877: ((t)-((-t)))+(((-(t)*(t)))+(199.6317))
Generation 68: Fitness 1799.493452431426: ((t)-((-t)))+(((-(t)*(t)))+(199.6319))
Generation 71: Fitness 1800.063773696427: ((t)-((-t)))+(((-(t)*(t)))+(199.6919))
Generation 73: Fitness 1800.069741481259: ((t)*((-t)))+(((t)-((-t)))+(199.9979))
Generation 74: Fitness 1777.364440211333: ((t)*((-t)))+(((t)*(2.7291475))+(196.5526))
Generation 79: Fitness 1778.363642363487: ((t)*((-t)))+(((t)*(2.7291975))+(196.5526))
Generation 81: Fitness 1777.468641476331: ((t)*((-t)))+(((t)*(2.9291475))+(196.5526))
Generation 82: Fitness 1777.667437525799: ((t)*((-t)))+(((t)*(2.9291475))+(196.5522))
Generation 87: Fitness 1778.578053235799: ((t)*((-t)))+(((t)*(2.9291475))+(196.5222))
Generation 88: Fitness 1776.641327759133: ((t)*((-t)))+(((t)*(2.9291475))+(194.8522))
Generation 95: Fitness 1778.041249694399: ((t)*((-t)))+(((t)*(2.9291475))+(194.8524))
Generation 98: Fitness 1778.641054882565: ((t)*((-t)))+(((t)*(2.9291475))+(194.8529))
Generation 101: Fitness 1779.239020617134: ((t)*((-t)))+(((t)*(2.9291475))+(194.8582))
Generation 103: Fitness 1780.756345699091: ((t)*((-t)))+(((t)*(2.9341475))+(194.8522))
Generation 105: Fitness 1781.467345037149: ((t)*((-t)))+(((t)*(2.9991475))+(194.8524))
Generation 106: Fitness 1782.267256501882: ((t)*((-t)))+(((t)*(2.9991475))+(194.8522))
Generation 108: Fitness 1783.858806975216: ((t)*((-t)))+(((t)*(2.9991475))+(194.8322))
Generation 114: Fitness 1788.119547860993: ((t)*((-t)))+(((t)*(Pi()))+(194.8522))
Generation 115: Fitness 1788.368275762677: ((t)*((-t)))+(((t)*(Pi()))+(194.5522))
Generation 118: Fitness 1790.397003664362: ((t)*((-t)))+(((t)*(Pi()))+(194.2522))
```

```
Generation 121: Fitness 1792.778652191141: ((t)*((-t)))+(((t)*(Pi()))+(194.2322))
Generation 122: Fitness 1793.544922036262: ((t)*((-t)))+(((t)*(Pi()))+(194.1929))
Generation 123: Fitness 1794.230702101627: ((t)*((-t)))+(((t)*(Pi()))+(193.5522))
Generation 126: Fitness 1796.582057104138: ((t)*((-t)))+(((t)*(Pi()))+(193.8528))
Generation 127: Fitness 1797.330102174856: ((t)*((-t)))+(((t)*(3.2))+(194.1323))
Generation 130: Fitness 1800.516273964024: ((t)*((-t)))+(((t)*(3.2))+(194.1223))
Generation 132: Fitness 1801.00785459744: ((t)*((-t)))+(((t)*(3.2))+(193.8393))
Generation 139: Fitness 1806.537281091524: ((t)*((-t)))+(((t)*(3.2))+(193.1323))
Generation 140: Fitness 1808.126048784215: ((t)*((-t)))+(((t)*(3.2))+(193.1514))
Generation 141: Fitness 1808.058642674415: ((t)*((-t)))+(((t)*(3.2))+(193.5522))
Generation 142: Fitness 1808.852564041914: ((t)*((-t)))+(((t)*(3.2))+(193.5222))
Generation 143: Fitness 1809.645380566082: ((t)*((-t)))+(((t)*(3.2))+(193.4522))
Generation 149: Fitness 1813.724261409072: ((t)*((-t)))+(((t)*(3.3))+(192.8525))
Generation 151: Fitness 1814.601062917081: ((t)*((-t)))+(((t)*(3.6))+(191.0562))
Generation 168: Fitness 1828.201062828647: ((t)*((-t)))+(((t)*(3.6))+(191.0558))
Generation 187: Fitness 1843.401062820756: ((t)*((-t)))+(((t)*(3.6))+(191.0559))
Generation 194: Fitness 1579.909903762592: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9142035))+(191.0558)))
Generation 198: Fitness 1584.811573168459: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9142235))+(191.0558)))
Generation 200: Fitness 1583.518251335492: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9342035))+(191.0558)))
Generation 209: Fitness 1501.919659868823: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9342035))+(199.0558)))
Generation 216: Fitness 1510.353606015801: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9352035))+(199.0558)))
Generation 218: Fitness 1512.021712688825: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9342035))+(199.3558)))
Generation 219: Fitness 1508.151452347957: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9742035))+(199.0558)))
Generation 225: Fitness 1512.55089725419: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9942035))+(199.0558)))
Generation 226: Fitness 1513.630531601168: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9952035))+(199.0558)))
Generation 231: Fitness 1519.801065574724: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9942235))+(199.0958)))
Generation 233: Fitness 1521.92341803846: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9972035))+(199.0558)))
Generation 235: Fitness 1523.186322380856: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9942035))+(199.9558)))
Generation 240: Fitness 1530.478149370596: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9947035))+(199.9558)))
Generation 241: Fitness 1530.546322380857: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9942035))+(199.9558)))
Generation 243: Fitness 1532.863333394502: ((t)*((-t)))+(((t)*((-t)))+(((t)*(9.9952035))+(199.9558)))
Generation 245: Fitness 1432.761782046678: ((t)*((-t)))+(((t)*((-t)))+(((t)*(Cosh(Pi())))+(199.0958)))
Generation 246: Fitness 1434.006221317489: ((t)*((-t)))+(((t)*((-t)))+(((t)*(12))+(199.0558)))
Generation 247: Fitness 1428.820833177492: ((t)*((-t)))+(((t)*((-t)))+(((t)*(11.3))+(199.0958)))
Generation 255: Fitness 1376.420270317491: ((t)*((-t)))+(((t)*((-t)))+(((t)*(12))+(195.0558)))
Generation 257: Fitness 1348.404140400825: ((t)*((-t)))+(((t)*((-t)))+(((t)*(12))+(192.0558)))
Generation 259: Fitness 1344.700985984159: ((t)*((-t)))+(((t)*((-t)))+(((t)*(12))+(191.0558)))
Generation 261: Fitness 1344.034777892492: ((t)*((-t)))+(((t)*((-t)))+(((t)*(12))+(190.3558)))
Generation 262: Fitness 1343.918849459159: ((t)*((-t)))+(((t)*((-t)))+(((t)*(12))+(189.9558)))
Generation 266: Fitness 1287.890022294501: ((t)*((-t)))+(((t)*((-t)))+(((t)*(13))+(179.0555)))
Generation 276: Fitness 1295.241399161167: ((t)*((-t)))+(((t)*((-t)))+(((t)*(13))+(179.8555)))
Generation 282: Fitness 1297.213345716161: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(179.0755)))
Generation 286: Fitness 1259.70076288239: ((t)*((-t)))+(((t)*((-t)))+(((t)*(14))+(179.0554)))
Generation 294: Fitness 1242.974468216164: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(169.0755)))
Generation 298: Fitness 1247.568798132829: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(172.0755)))
Generation 299: Fitness 1246.899577049496: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(171.0755)))
Generation 304: Fitness 1253.01539262783: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(171.0555)))
Generation 307: Fitness 1256.69505043273: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(171.0551)))
Generation 309: Fitness 1258.987477391164: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(170.7755)))
Generation 311: Fitness 1261.602059294496: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(171.0555)))
Generation 315: Fitness 1266.328299841163: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15))+(170.5755)))
Generation 316: Fitness 1266.590412290598: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.26629))+(170.0755)))
Generation 320: Fitness 1271.658112098932: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.26629))+(169.0555)))
Generation 323: Fitness 1274.107220916398: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.26729))+(169.0555)))
Generation 330: Fitness 1283.920550595857: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.26729))+(169.0554)))
Generation 334: Fitness 1287.592389627999: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.26929))+(169.0555)))
Generation 336: Fitness 1290.020133090265: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.27629))+(169.0555)))
Generation 337: Fitness 1291.197902323732: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29729))+(169.0555)))
Generation 339: Fitness 1293.700133090265: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.27629))+(169.0555)))
Generation 340: Fitness 1296.124967748265: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.28629))+(169.0555)))
Generation 341: Fitness 1296.105949406265: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29629))+(169.0555)))
Generation 346: Fitness 1302.238902000848: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29629))+(169.0545)))
Generation 348: Fitness 1304.691369432501: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29719))+(169.0555)))
Generation 349: Fitness 1305.914138655446: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29724))+(169.0455)))
Generation 350: Fitness 1308.362117497619: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29626))+(169.0255)))
Generation 356: Fitness 1314.491979117515: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29629))+(169.0145)))
Generation 359: Fitness 1318.170695859066: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29714))+(169.0155)))
Generation 361: Fitness 1320.623725419857: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29729))+(169.0154)))
Generation 367: Fitness 1327.980328351998: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29929))+(169.0155)))
Generation 368: Fitness 1329.208661827865: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29829))+(169.0155)))
Generation 369: Fitness 1330.433325246581: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29929))+(169.0145)))
Generation 371: Fitness 1332.883720631165: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29929))+(169.0055)))
Generation 376: Fitness 1339.015860306041: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(169.0055)))
Generation 388: Fitness 1353.735827739166: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(169.0054)))
Generation 390: Fitness 1357.415408589501: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29969))+(169.0025)))
Generation 398: Fitness 1366.001239897708: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(169.0015)))
Generation 408: Fitness 1378.267874797499: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(169.0014)))
Generation 409: Fitness 1379.494477990417: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(169.0012)))
Generation 431: Fitness 1406.480860015208: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(169.0003)))
Generation 433: Fitness 1408.934130314791: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(169.0001)))
Generation 486: Fitness 1473.942973023125: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(168.7001)))
Generation 499: Fitness 1489.872633439792: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.29999))+(168.9001)))
Generation 509: Fitness 1502.099692503125: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.7001)))
Generation 510: Fitness 1504.550538484178: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39939))+(168.7001)))
```

```
Generation 511: Fitness 1504.553025836458: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.7001)))
Generation 515: Fitness 1509.422692388458: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.38999))+(168.7001)))
Generation 518: Fitness 1513.059195628125: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.6001)))
Generation 520: Fitness 1516.729004013208: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.35999))+(168.7011)))
Generation 521: Fitness 1516.739195628125: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.6001)))
Generation 523: Fitness 1519.181907377791: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.35999))+(168.7001)))
Generation 525: Fitness 1521.523780763542: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2011)))
Generation 530: Fitness 1528.881659513541: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2411)))
Generation 531: Fitness 1528.883780763542: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2011)))
Generation 533: Fitness 1531.336841284375: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2911)))
Generation 534: Fitness 1532.561673544791: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2401)))
Generation 538: Fitness 1537.468295867708: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2511)))
Generation 545: Fitness 1546.054956565625: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2501)))
Generation 546: Fitness 1547.281617326042: ((t)*((-t)))+(((t)*((-t)))+(((t)*(15.39999))+(168.2471)))
Generation 553: Fitness 1479.551160643221: (((t)-((-t)))*((-t)))+(((t)*(15.39959))+(168.2011))
Generation 557: Fitness 1483.817817540763: (((t)-((-t)))*((-t)))+(((t)*(15.39959))+(168.2012))
Generation 564: Fitness 1491.28218055743: (((t)-((-t)))*((-t)))+(((t)*(15.39959))+(168.2412))
Generation 570: Fitness 1498.748842036721: (((t)-((-t)))*((-t)))+(((t)*(15.39959))+(168.2415))
Generation 575: Fitness 1503.01525082619: (((t)-((-t)))*((-t)))+(((t)*(15.39979))+(168.2412))
Generation 579: Fitness 1507.281658220417: (((t)-((-t)))*((-t)))+(((t)*(15.39999))+(168.2412))
Generation 587: Fitness 1515.814987794375: (((t)-((-t)))*((-t)))+(((t)*(15.39999))+(168.2415))
Generation 589: Fitness 1517.948284068334: (((t)-((-t)))*((-t)))+(((t)*(15.39999))+(168.2482))
Generation 590: Fitness 1520.081617232917: (((t)-((-t)))*((-t)))+(((t)*(15.39999))+(168.2472))
Generation 597: Fitness 1526.481617073542: (((t)-((-t)))*((-t)))+(((t)*(15.39999))+(168.2475))
Generation 603: Fitness 1532.881617060416: (((t)-((-t)))*((-t)))+(((t)*(15.39999))+(168.2476))
Generation 619: Fitness 1492.028283899584: (((t)+(t))*((-t)))+(((t)*(15.39999))+(168.2472))
Generation 630: Fitness 1503.548283740208: (((t)+(t))*((-t)))+(((t)*(15.39999))+(168.2475))
Generation 635: Fitness 1507.388283733958: (((t)+(t))*((-t)))+(((t)*(15.39999))+(168.2477))
Generation 637: Fitness 1509.308283727083: (((t)+(t))*((-t)))+(((t)*(15.39999))+(168.2476))
Generation 683: Fitness 1335.490071602587: ((((t)+(t))+(t))*((-t)))+(((t)*(25.79999))+(168.2476))
Generation 689: Fitness 1341.670494250053: ((((t)+(t))+(t))*((-t)))+(((t)*(25.79599))+(168.2476))
Generation 691: Fitness 1344.450071602586: ((((t)+(t))+(t))*((-t)))+(((t)*(25.79999))+(168.2476))
Generation 695: Fitness 1338.426726685253: ((((t)+(t))+(t))*((-t)))+(((t)*(25.71999))+(168.2476))
Generation 698: Fitness 1271.728848669258: ((((t)+(t))+(t))*((-t)))+(((t)*(23.79999))+(168.2476))
Generation 701: Fitness 1201.953623685921: ((((t)+(t))+(t))*((-t)))+(((t)*(25.79999))+(161.2476))
Generation 703: Fitness 1178.757363269257: ((((t)+(t))+(t))*((-t)))+(((t)*(25.79999))+(148.2476))
Generation 706: Fitness 1171.683717435922: ((((t)+(t))+(t))*((-t)))+(((t)*(25.79999))+(158.2476))
Generation 707: Fitness 1172.803395775986: ((((t)+(t))+(t))*((-t)))+(((t)*(25.79997))+(158.2476))
Generation 708: Fitness 1128.842974735922: ((((t)+(t))+(t))*((-t)))+(((t)*(26.79999))+(148.2476))
Generation 721: Fitness 1143.394508381755: ((((t)+(t))+(t))*((-t)))+(((t)*(26.79999))+(148.2376))
Generation 723: Fitness 1145.305719215922: ((((t)+(t))+(t))*((-t)))+(((t)*(26.89999))+(148.2476))
Generation 729: Fitness 1152.025678230239: ((((t)+(t))+(t))*((-t)))+(((t)*(26.89996))+(148.2476))
Generation 730: Fitness 1154.245352861755: ((((t)+(t))+(t))*((-t)))+(((t)*(26.89999))+(148.2376))
Generation 733: Fitness 1155.459399813572: ((((t)+(t))+(t))*((-t)))+(((t)*(26.89996))+(147.2476))
Generation 735: Fitness 1157.119861612588: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(147.2476))
Generation 736: Fitness 1158.239737959047: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(147.2475))
Generation 737: Fitness 1159.123226195921: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(146.2476))
Generation 742: Fitness 1164.577880362589: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(146.6476))
Generation 750: Fitness 1174.657873075505: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(146.6474))
Generation 754: Fitness 1178.017696550088: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(146.6416))
Generation 755: Fitness 1179.137547654255: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(146.6276))
Generation 765: Fitness 1190.337544925088: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(146.6296))
Generation 782: Fitness 1209.377544824463: ((((t)+(t))+(t))*((-t)))+(((t)*(26.99999))+(146.6293))
Generation 798: Fitness 1225.267462786423: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.99999))+(146.6292))
Generation 805: Fitness 1234.047727756824: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.99799))+(146.6292))
Generation 810: Fitness 1239.251254742424: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.96999))+(146.6292))
Generation 813: Fitness 1241.715912734091: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.96999))+(146.6252))
Generation 819: Fitness 1155.238402317425: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.96999))+(141.6252))
Generation 824: Fitness 1117.496233909091: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.96999))+(136.6292))
Generation 835: Fitness 1130.957558938425: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.98999))+(136.6292))
Generation 836: Fitness 1130.016731825758: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.96999))+(135.6292))
Generation 839: Fitness 1133.017229742425: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.96999))+(134.6292))
Generation 846: Fitness 1141.977149743467: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.96999))+(134.6297))
Generation 847: Fitness 1142.705152090425: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(36.97999))+(134.6292))
Generation 849: Fitness 1126.797612459091: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(133.6292))
Generation 858: Fitness 1138.102873754024: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96199))+(133.6292))
Generation 859: Fitness 1137.349473979091: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.86999))+(133.6292))
Generation 864: Fitness 1137.258110375757: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(132.6292))
Generation 866: Fitness 1133.078608292424: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.6292))
Generation 871: Fitness 1137.304836492624: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96899))+(131.2212))
Generation 881: Fitness 1149.658857250758: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.1292))
Generation 884: Fitness 1153.465729736174: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.1222))
Generation 890: Fitness 1162.411562229925: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.1192))
Generation 891: Fitness 1162.425729736174: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.1222))
Generation 894: Fitness 1166.261005234091: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.1212))
Generation 896: Fitness 1168.792455432174: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.98999))+(131.1222))
Generation 897: Fitness 1170.070038780174: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(131.1222))
Generation 898: Fitness 1171.338574715341: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.1122))
Generation 906: Fitness 1181.573870213258: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.96999))+(131.1112))
Generation 909: Fitness 1185.379313759341: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(131.1122))
Generation 914: Fitness 1187.999850334175: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.97999))+(130.1222))
Generation 920: Fitness 1196.707536696841: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(130.1222))
Generation 928: Fitness 1205.664455194758: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(130.1212))
Generation 935: Fitness 1214.596811676008: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(130.1122))
```

```
Generation 945: Fitness 1227.393750173924: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(130.1112))
Generation 974: Fitness 1264.217699965591: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(130.0112))
Generation 975: Fitness 1263.713633405262: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(129.09237))
Generation 987: Fitness 1279.055268039651: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99992))+(129.07237))
Generation 989: Fitness 1281.594319721929: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(129.05237))
Generation 990: Fitness 1284.170791321929: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(128.09237))
Generation 992: Fitness 1285.415862880262: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(129.03237))
Generation 995: Fitness 1289.239947494651: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99992))+(129.01237))
Generation 997: Fitness 1291.798206038596: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(129.01237))
Generation 1005: Fitness 1301.887602471658: ((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(128.39248))
Generation 1008: Fitness 1305.694303679651: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99992))+(128.59237))
Generation 1018: Fitness 1318.492212363596: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(37.99999))+(128.59237))
Generation 1031: Fitness 1330.767509308747: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.4))+(128.59237))
Generation 1036: Fitness 1335.822620983747: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.4))+(128.29237))
Generation 1040: Fitness 1341.42602876708: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.4))+(128.09237))
Generation 1051: Fitness 1350.660106600413: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.4))+(126.09237))
Generation 1054: Fitness 1351.061662808747: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.6))+(124.59237))
Generation 1058: Fitness 1354.566119508747: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.7))+(124.39237))
Generation 1063: Fitness 1360.359793975413: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.8))+(124.59237))
Generation 1072: Fitness 1371.611201758747: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.8))+(124.39237))
Generation 1073: Fitness 1372.70260954208: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.8))+(124.19237))
Generation 1074: Fitness 1372.888060150413: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.29237))
Generation 1083: Fitness 1384.40683053958: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.28237))
Generation 1088: Fitness 1390.80365287458: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.22237))
Generation 1094: Fitness 1398.48364075783: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.22537))
Generation 1104: Fitness 1411.283640718913: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.22637))
Generation 1110: Fitness 1420.243640536155: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.22567))
Generation 1113: Fitness 1422.803640488371: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.22587))
Generation 1121: Fitness 1433.043640487993: (((((t)+(t))+(t))+(t))*((-t)))+(((t)*(38.9))+(123.22589))
Generation 1153: Fitness 1417.670948540286: ((((t)+(Exp(ArcSinh(t))))+(t))*((-t)))+(((t)*(38.9))+(123.22589))
Generation 1167: Fitness 1434.639154347161: ((((t)+(Exp(ArcSinh(t))))+(t))*((-t)))+(((t)*(38.9))+(123.82589))
Generation 1171: Fitness 1439.530008934904: ((((t)+(Exp(ArcSinh(t))))+(t))*((-t)))+(((t)*(38.9))+(123.72589))
Generation 1179: Fitness 1449.342617436668: ((((t)+(Exp(ArcSinh(t))))+(t))*((-t)))+(((t)*(38.9))+(123.68589))
Generation 1186: Fitness 1457.929165311227: ((((t)+(Exp(ArcSinh(t))))+(t))*((-t)))+(((t)*(38.9))+(123.69589))
Generation 1196: Fitness 1470.195831799972: ((((t)+(Exp(ArcSinh(t))))+(t))*((-t)))+(((t)*(38.9))+(123.69599))
Generation 1201: Fitness 1476.329164432016: ((((t)+(Exp(ArcSinh(t))))+(t))*((-t)))+(((t)*(38.9))+(123.69689))
Generation 1208: Fitness 1423.454148306058: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(38.9))+(123.68589))
Generation 1217: Fitness 1431.07345741556: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(38.9))+(123.88589))
Generation 1220: Fitness 1398.993421591111: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(38.9))+(129.68589))
Generation 1221: Fitness 1417.007239401076: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(38.9))+(125.68589))
Generation 1222: Fitness 1410.623784948585: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(38.9))+(126.68589))
Generation 1223: Fitness 1402.736926113091: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(38.9))+(128.68587))
Generation 1227: Fitness 1368.88841854365: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.69599))
Generation 1230: Fitness 1373.306393255747: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.83589))
Generation 1232: Fitness 1374.48841854365: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.69599))
Generation 1234: Fitness 1376.653720533122: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.88589))
Generation 1240: Fitness 1384.491767653152: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.89599))
Generation 1245: Fitness 1388.963374752428: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.98599))
Generation 1254: Fitness 1399.043372243523: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.98799))
Generation 1255: Fitness 1400.16337398907: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.98899))
Generation 1258: Fitness 1403.523372243523: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(39.9))+(123.98799))
Generation 1266: Fitness 1410.470703397517: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.98599))
Generation 1270: Fitness 1416.070228859298: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.98559))
Generation 1271: Fitness 1416.065972415327: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.98199))
Generation 1273: Fitness 1418.305147783444: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.98129))
Generation 1275: Fitness 1420.5450536022: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.98121))
Generation 1277: Fitness 1422.677413228693: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.88129))
Generation 1278: Fitness 1423.55866237766: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38553))
Generation 1280: Fitness 1426.918659582769: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38573))
Generation 1282: Fitness 1428.038656867879: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38593))
Generation 1284: Fitness 1430.278656069012: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38599))
Generation 1288: Fitness 1434.75862757199: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38869))
Generation 1294: Fitness 1441.478621776089: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39572))
Generation 1295: Fitness 1442.598625744522: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38893))
Generation 1296: Fitness 1443.71862048496: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38973))
Generation 1298: Fitness 1445.958620633135: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39553))
Generation 1299: Fitness 1447.078619370069: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.38993))
Generation 1303: Fitness 1451.558616612697: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39473))
Generation 1306: Fitness 1454.918613872259: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39393))
Generation 1308: Fitness 1457.15861300633: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39169))
Generation 1309: Fitness 1458.278612246712: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39293))
Generation 1310: Fitness 1460.518612156493: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39253))
Generation 1319: Fitness 1469.47861215418: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39269))
Generation 1326: Fitness 1477.318612149626: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39259))
Generation 1327: Fitness 1478.438612149047: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39263))
Generation 1332: Fitness 1484.038612148892: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40))+(123.39262))
Generation 1452: Fitness 1616.692612290594: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(41))+(120.39262))
Generation 1460: Fitness 1626.596512124168: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(41))+(120.36262))
Generation 1463: Fitness 1623.8647367385: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.81558))+(120.39262))
Generation 1470: Fitness 1632.76764795348: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.81258))+(120.39262))
Generation 1474: Fitness 1636.108806954297: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.81158))+(120.39262))
Generation 1478: Fitness 1640.374704134699: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.81558))+(120.31262))
Generation 1479: Fitness 1639.846790573919: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61558))+(120.39292))
Generation 1480: Fitness 1642.086392861655: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61558))+(120.39262))
```

```
Generation 1482: Fitness 1643.2062603309: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61558))+(120.39252))
Generation 1483: Fitness 1644.227109970118: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61558))+(120.31292))
Generation 1489: Fitness 1650.923871698143: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61556))+(120.29262))
Generation 1491: Fitness 1653.163739576149: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61558))+(120.29252))
Generation 1493: Fitness 1655.301589064417: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61558))+(120.19292))
Generation 1498: Fitness 1660.901567418718: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61559))+(120.19292))
Generation 1499: Fitness 1662.004678791593: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.62558))+(120.19292))
Generation 1500: Fitness 1664.193908460617: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61558))+(120.11292))
Generation 1506: Fitness 1669.792977613582: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61588))+(120.11292))
Generation 1508: Fitness 1671.930579375373: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.61258))+(119.89))
Generation 1509: Fitness 1673.129925008005: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65558))+(120.13262))
Generation 1512: Fitness 1676.440998491193: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.64558))+(120.09292))
Generation 1515: Fitness 1679.739886381286: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.63558))+(120.01262))
Generation 1519: Fitness 1684.201388690043: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65558))+(120.02292))
Generation 1520: Fitness 1686.459484927512: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.63558))+(120.01212))
Generation 1521: Fitness 1686.451784253254: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65558))+(120.03262))
Generation 1522: Fitness 1687.559531068919: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.67558))+(120.01292))
Generation 1523: Fitness 1688.660242026829: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65558))+(120.00262))
Generation 1526: Fitness 1692.018756199981: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00262))
Generation 1539: Fitness 1706.578622556002: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65898))+(120.00262))
Generation 1540: Fitness 1708.818450531033: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65958))+(120.00262))
Generation 1541: Fitness 1708.818227896208: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00212))
Generation 1548: Fitness 1716.657700092434: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00162))
Generation 1549: Fitness 1717.777561688454: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65898))+(120.00162))
Generation 1551: Fitness 1720.017700092434: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00162))
Generation 1554: Fitness 1723.37738365017: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00132))
Generation 1556: Fitness 1725.61717278866: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00112))
Generation 1565: Fitness 1735.69706251122: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65958))+(120.00132))
Generation 1567: Fitness 1737.936921452009: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66258))+(120.00155))
Generation 1570: Fitness 1742.416849269711: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65958))+(120.00112))
Generation 1571: Fitness 1742.416224901867: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00022))
Generation 1572: Fitness 1743.536119681113: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00012))
Generation 1576: Fitness 1748.016014480358: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65858))+(120.00002))
Generation 1579: Fitness 1751.375939574301: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65878))+(120.00002))
Generation 1581: Fitness 1753.615890672918: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65958))+(120.00022))
Generation 1582: Fitness 1754.735677871409: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.65958))+(120.00002))
Generation 1583: Fitness 1755.855446216904: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66251))+(120.00021))
Generation 1595: Fitness 1769.295446052414: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66253))+(120.00021))
Generation 1597: Fitness 1771.535445786473: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66259))+(120.00021))
Generation 1598: Fitness 1772.655336321448: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66251))+(120.00011))
Generation 1599: Fitness 1773.775336133159: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66253))+(120.00011))
Generation 1609: Fitness 1784.975335759214: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66261))+(120.00011))
Generation 1626: Fitness 1804.015214766413: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66261))+(120))
Generation 1627: Fitness 1805.135335759214: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66261))+(120.00011))
Generation 1628: Fitness 1806.255335755132: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66262))+(120.00011))
Generation 1631: Fitness 1809.615225764759: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66261))+(120.00001))
Generation 1632: Fitness 1810.735225742276: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66263))+(120.00001))
Generation 1705: Fitness 1892.395143194767: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66263))+(119.00001))
Generation 1716: Fitness 1904.600881600276: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.67263))+(119.00001))
Generation 1717: Fitness 1905.695159704269: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.66263))+(119.20001))
Generation 1721: Fitness 1910.096100339118: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.68263))+(119.00001))
Generation 1723: Fitness 1912.240799411294: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.69263))+(119.00001))
Generation 1724: Fitness 1912.959142249857: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.76263))+(119.00001))
Generation 1725: Fitness 1914.068228077611: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.76763))+(119.00001))
Generation 1732: Fitness 1921.899765845208: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.77263))+(119.00021))
Generation 1735: Fitness 1925.249340009699: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.79293))+(119.00001))
Generation 1739: Fitness 1929.729208466383: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.79263))+(119.00001))
Generation 1740: Fitness 1931.75109209077: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.86263))+(118.00001))
Generation 1741: Fitness 1931.647932637487: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89263))+(118.00401))
Generation 1747: Fitness 1938.367544971665: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89283))+(118.00401))
Generation 1748: Fitness 1939.48251582414: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89563))+(118.00401))
Generation 1749: Fitness 1940.607352560803: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89293))+(118.00401))
Generation 1751: Fitness 1942.842237590338: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89263))+(118.06401))
Generation 1753: Fitness 1945.082090066763: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89263))+(118.09401))
Generation 1754: Fitness 1946.200899826358: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89663))+(118.00401))
Generation 1755: Fitness 1947.316614930014: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89963))+(118.00409))
Generation 1759: Fitness 1951.79319470663: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.90263))+(118.00401))
Generation 1760: Fitness 1954.036614930014: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89963))+(118.00409))
Generation 1762: Fitness 1955.155355954911: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89963))+(118.04401))
Generation 1776: Fitness 1970.835325888634: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89968))+(118.03401))
Generation 1778: Fitness 1973.075278714276: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89993))+(118.02401))
Generation 1782: Fitness 1977.555196213754: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89983))+(118.04401))
Generation 1783: Fitness 1978.67509823951: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89993))+(118.03401))
Generation 1784: Fitness 1979.79509815937 6: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89993))+(118.03402))
Generation 1787: Fitness 1983.155083002488: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89993))+(118.03901))
Generation 1790: Fitness 1987.635044412455: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03402))
Generation 1797: Fitness 1994.35504248735: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.04106))
Generation 1799: Fitness 1996.595038640348: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03492))
Generation 1802: Fitness 1999.955032818428: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03901))
Generation 1804: Fitness 2002.195031258642: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03721))
Generation 1806: Fitness 2004.43503115188: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03801))
Generation 1814: Fitness 2013.395031041916: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03771))
Generation 1819: Fitness 2018.995031040854: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03767))
Generation 1832: Fitness 2033.555031040819: ((((t)*(Pi()))+(t))*((-t)))+(((t)*(40.89999))+(118.03768))
```

```
Generation 2116: Fitness 2165.152354969269: ((((t)*(3.162))+(t))*((-t)))+(((t)*(40.89999))+(118.03768))
Generation 2130: Fitness 2165.039874294269: ((((t)*(3.162))+(t))*((-t)))+(((t)*(40.89999))+(118.09768))
Generation 2133: Fitness 2163.807502649269: ((((t)*(3.162))+(t))*((-t)))+(((t)*(40.99999))+(118.03768))
Generation 2137: Fitness 2163.776228107602: ((((t)*(3.162))+(t))*((-t)))+(((t)*(40.99999))+(118.73768))
Generation 2138: Fitness 2163.368225848357: ((((t)*(3.172))+(t))*((-t)))+(((t)*(40.99999))+(118.03768))
Generation 2142: Fitness 2163.069683598357: ((((t)*(3.172))+(t))*((-t)))+(((t)*(40.99999))+(118.23768))
Generation 2146: Fitness 2162.851141348357: ((((t)*(3.172))+(t))*((-t)))+(((t)*(40.99999))+(118.43768))
Generation 2150: Fitness 2162.654785723357: ((((t)*(3.172))+(t))*((-t)))+(((t)*(40.99999))+(118.93768))
Generation 2151: Fitness 2162.656086610731: ((((t)*(3.172))+(t))*((-t)))+(((t)*(40.99979))+(118.93768))
Generation 2152: Fitness 2162.500847852953: ((((t)*(3.175))+(t))*((-t)))+(((t)*(40.99999))+(118.73768))
Generation 2154: Fitness 2162.174619006237: ((((t)*(3.182))+(t))*((-t)))+(((t)*(40.99999))+(118.93758))
Generation 2159: Fitness 2162.171277486376: ((((t)*(3.183))+(t))*((-t)))+(((t)*(40.99999))+(118.93758))
Generation 2164: Fitness 2162.142581222904: ((((t)*(3.182))+(t))*((-t)))+(((t)*(40.99999))+(118.97758))
Generation 2167: Fitness 2162.100460463446: ((((t)*(3.182))+(t))*((-t)))+(((t)*(40.99999))+(119.03768))
Generation 2176: Fitness 2040.936766073253: (((t)*(3.972))*((-t)))+(((t)*(40.99999))+(118.83768))
Generation 2181: Fitness 2161.692027295672: ((((t)*(3.194))+(t))*((-t)))+(((t)*(40.99999))+(119.93768))
Generation 2192: Fitness 2161.692026217401: ((((t)*(3.194))+(t))*((-t)))+(((t)*(40.99999))+(119.93763))
Generation 2195: Fitness 2161.692026002347: ((((t)*(3.194))+(t))*((-t)))+(((t)*(40.99999))+(119.93762))
Generation 2196: Fitness 2161.691971449422: ((((t)*(3.194))+(t))*((-t)))+(((t)*(40.99999))+(119.93468))
Generation 2198: Fitness 2161.691933603171: ((((t)*(3.194))+(t))*((-t)))+(((t)*(40.99999))+(119.93168))
Generation 2200: Fitness 2161.691343682064: ((((t)*(3.195))+(t))*((-t)))+(((t)*(40.99999))+(119.93768))
Generation 2201: Fitness 2161.691933603171: ((((t)*(3.194))+(t))*((-t)))+(((t)*(40.99999))+(119.93168))
Generation 2204: Fitness 2161.691343682064: ((((t)*(3.195))+(t))*((-t)))+(((t)*(40.99999))+(119.93768))
Generation 2211: Fitness 2161.691336373272: ((((t)*(3.195))+(t))*((-t)))+(((t)*(40.99999))+(119.93778))
Generation 2213: Fitness 2161.690552407064: ((((t)*(3.195))+(t))*((-t)))+(((t)*(40.99999))+(119.99768))
Generation 2216: Fitness 2155.419629853872: ((((t)*(3.594))+(t))*((-t)))+(((t)*(44.99999))+(119.93768))
Generation 2225: Fitness 2155.419047840547: ((((t)*(3.594))+(t))*((-t)))+(((t)*(44.99999))+(119.93762))
Generation 2227: Fitness 2155.417689837455: ((((t)*(3.594))+(t))*((-t)))+(((t)*(44.99999))+(119.93748))
Generation 2230: Fitness 2141.817373187206: ((((t)*(3.594))+(t))*((-t)))+(((t)*(44.99999))+(111.93768))
Generation 2231: Fitness 2152.578857387214: ((((t)*(3.594))+(t))*((-t)))+(((t)*(44.59999))+(119.93762))
Generation 2234: Fitness 2024.023604235286: (((t)*(3.995))*((-t)))+(((t)*(40.99992))+(119.96768))
Generation 2239: Fitness 2010.354487284286: (((t)*(3.995))*((-t)))+(((t)*(40.90992))+(119.96768))
Generation 2241: Fitness 2024.023604235286: (((t)*(3.995))*((-t)))+(((t)*(40.99992))+(119.96768))
Generation 2243: Fitness 2022.940684525018: (((t)*(3.995))*((-t)))+(((t)*(40.99297))+(119.96768))
Generation 2246: Fitness 1929.444489115287: (((t)*(3.995))*((-t)))+(((t)*(40.19992))+(119.96768))
Generation 2256: Fitness 1927.078073798287: (((t)*(3.995))*((-t)))+(((t)*(40.16992))+(119.96768))
Generation 2258: Fitness 1926.70649539287: (((t)*(3.995))*((-t)))+(((t)*(40.19992))+(119.66768))
Generation 2260: Fitness 1912.413953807821: (((t)*(3.995))*((-t)))+(((t)*(40.23997))+(112.96768))
Generation 2263: Fitness 1910.166336043553: (((t)*(3.995))*((-t)))+(((t)*(40.19996))+(116.96768))
Generation 2264: Fitness 1908.884614615288: (((t)*(3.995))*((-t)))+(((t)*(40.19992))+(113.96768))
Generation 2266: Fitness 1907.735268639571: (((t)*(3.995))*((-t)))+(((t)*(40.19997))+(115.96468))
Generation 2267: Fitness 1906.913926504346: (((t)*(3.995))*((-t)))+(((t)*(40.17992))+(114.96798))
Generation 2269: Fitness 1906.744377310557: (((t)*(3.998))*((-t)))+(((t)*(40.23997))+(114.96768))
Generation 2274: Fitness 1905.924179205488: (((t)*(3.995))*((-t)))+(((t)*(40.11792))+(114.96768))
Generation 2275: Fitness 1905.49554368759: (((t)*(3.999))*((-t)))+(((t)*(40.19992))+(114.96768))
Generation 2276: Fitness 1905.924179205488: (((t)*(3.995))*((-t)))+(((t)*(40.11792))+(114.96768))
Generation 2277: Fitness 1905.680995042288: (((t)*(3.995))*((-t)))+(((t)*(40.12992))+(115.96768))
Generation 2278: Fitness 1905.542976104257: (((t)*(3.999))*((-t)))+(((t)*(40.19992))+(115.96768))
Generation 2281: Fitness 1905.522476195752: (((t)*(3.999))*((-t)))+(((t)*(40.19922))+(115.96767))
Generation 2282: Fitness 1905.415979021622: (((t)*(3.995))*((-t)))+(((t)*(40.11992))+(115.94768))
Generation 2283: Fitness 1905.163728507409: (((t)*(3.999))*((-t)))+(((t)*(40.17992))+(114.96995))
Generation 2284: Fitness 1905.402120721621: (((t)*(3.995))*((-t)))+(((t)*(40.11992))+(115.54768))
Generation 2285: Fitness 1904.155681380517: (((t)*(3.998))*((-t)))+(((t)*(40.12137))+(116.26768))
Generation 2289: Fitness 1903.724530050098: (((t)*(3.999))*((-t)))+(((t)*(40.12997))+(115.96498))
Generation 2292: Fitness 1903.289576753257: (((t)*(3.999))*((-t)))+(((t)*(40.10992))+(115.96768))
Generation 2293: Fitness 1903.18690369209: (((t)*(3.999))*((-t)))+(((t)*(40.10492))+(115.96768))
Generation 2296: Fitness 1896.473555731955: (((t)*(4.0545158))*((-t)))+(((t)*(40.11792))+(114.96768))
Generation 2305: Fitness 1883.707753827954: (((t)*(4.0545158))*((-t)))+(((t)*(40.11792))+(118.96768))
Generation 2312: Fitness 1878.474060424157: (((t)*(4.0845158))*((-t)))+(((t)*(40.11792))+(118.96768))
Generation 2320: Fitness 1877.82109922109: (((t)*(4.0845158))*((-t)))+(((t)*(40.16792))+(118.96768))
Generation 2326: Fitness 1877.405146351357: (((t)*(4.0845158))*((-t)))+(((t)*(40.21792))+(118.96768))
Generation 2327: Fitness 1877.28426561189: (((t)*(4.0845158))*((-t)))+(((t)*(40.31792))+(118.96768))
Generation 2330: Fitness 1877.268707008879: (((t)*(4.0845858))*((-t)))+(((t)*(40.31792))+(118.96768))
Generation 2334: Fitness 1876.368471556571: (((t)*(4.0945158))*((-t)))+(((t)*(40.51796))+(118.96768))
Generation 2340: Fitness 1875.464893142624: (((t)*(4.0945158))*((-t)))+(((t)*(40.31792))+(118.96768))
Generation 2345: Fitness 1875.34959445833: (((t)*(4.0945158))*((-t)))+(((t)*(40.38792))+(118.96768))
Generation 2348: Fitness 1875.077471321714: (((t)*(4.0975158))*((-t)))+(((t)*(40.31792))+(118.96768))
Generation 2349: Fitness 1874.913551389688: (((t)*(4.0995158))*((-t)))+(((t)*(40.51796))+(118.96768))
Generation 2354: Fitness 1874.913453885247: (((t)*(4.0995158))*((-t)))+(((t)*(40.51795))+(118.96768))
Generation 2355: Fitness 1874.886524239148: (((t)*(4.0996158))*((-t)))+(((t)*(40.51796))+(118.96768))
Generation 2357: Fitness 1868.830334095064: (((t)*(4.1945158))*((-t)))+(((t)*(40.81792))+(118.96761))
Generation 2360: Fitness 1874.398769901658: (((t)*(4.09982))*((-t)))+(((t)*(40.38792))+(118.96768))
Generation 2369: Fitness 1874.38369788945: (((t)*(4.0999158))*((-t)))+(((t)*(40.38792))+(118.96768))
Generation 2373: Fitness 1874.383070199138: (((t)*(4.0999198))*((-t)))+(((t)*(40.38792))+(118.96768))
Generation 2378: Fitness 1874.357128274336: (((t)*(4.0998158))*((-t)))+(((t)*(40.41796))+(118.96768))
Generation 2381: Fitness 1874.375058060131: (((t)*(4.09982))*((-t)))+(((t)*(40.39797))+(118.96768))
Generation 2382: Fitness 1874.371168775802: (((t)*(4.0999958))*((-t)))+(((t)*(40.38792))+(118.96768))
Generation 2383: Fitness 1874.357128274336: (((t)*(4.0998158))*((-t)))+(((t)*(40.41796))+(118.96768))
Generation 2386: Fitness 1874.356395239855: (((t)*(4.0998198))*((-t)))+(((t)*(40.41796))+(118.96768))
Generation 2390: Fitness 1874.338848802539: (((t)*(4.0999158))*((-t)))+(((t)*(40.41786))+(118.96768))
Generation 2391: Fitness 1874.34614640302: (((t)*(4.0998758))*((-t)))+(((t)*(40.41796))+(118.96768))
Generation 2395: Fitness 1874.345945161688: (((t)*(4.0998998))*((-t)))+(((t)*(40.42796))+(118.96768))
Generation 2396: Fitness 1874.34539842303: (((t)*(4.0999998))*((-t)))+(((t)*(40.39792))+(118.96768))
Generation 2399: Fitness 1874.338491629643: (((t)*(4.0999158))*((-t)))+(((t)*(40.41796))+(118.96268))
```

```
Generation 2407: Fitness 1874.337945700416: (((t)*(4.0999188))*((-t)))+(((t)*(40.41796))+(118.96268))
Generation 2408: Fitness 1874.327908264641: (((t)*(4.0999758))*((-t)))+(((t)*(40.41796))+(118.96768))
Generation 2410: Fitness 1874.323978089832: (((t)*(4.1))*((-t)))+(((t)*(40.41596))+(118.96768))
Generation 2416: Fitness 1874.323541858385: (((t)*(4.0999998))*((-t)))+(((t)*(40.41796))+(118.96766))
Generation 2418: Fitness 1874.323233244246: (((t)*(4.0999998))*((-t)))+(((t)*(40.41796))+(118.96268))
Generation 2425: Fitness 1874.322603383586: (((t)*(4.0999998))*((-t)))+(((t)*(40.41796))+(118.94768))
Generation 2426: Fitness 1874.322433337064: (((t)*(4.0999998))*((-t)))+(((t)*(40.41796))+(118.93766))
Generation 2427: Fitness 1874.320077321919: (((t)*(4.0999998))*((-t)))+(((t)*(40.42796))+(118.91768))
Generation 2430: Fitness 1874.319620711092: (((t)*(4.1))*((-t)))+(((t)*(40.42526))+(118.86768))
Generation 2431: Fitness 1874.320755393233: (((t)*(4.0999998))*((-t)))+(((t)*(40.42196))+(118.92768))
Generation 2432: Fitness 1874.320509486127: (((t)*(4.0999998))*((-t)))+(((t)*(40.42196))+(118.91768))
Generation 2437: Fitness 1874.319966956939: (((t)*(4.0999998))*((-t)))+(((t)*(40.42736))+(118.91768))
Generation 2443: Fitness 1874.319842163321: (((t)*(4.0999998))*((-t)))+(((t)*(40.42596))+(118.91768))
Generation 2449: Fitness 1874.319839504195: (((t)*(4.0999998))*((-t)))+(((t)*(40.42576))+(118.91768))
Generation 2451: Fitness 1874.319667554097: (((t)*(4.0999998))*((-t)))+(((t)*(40.42736))+(118.91368))
Generation 2452: Fitness 1874.319505019057: (((t)*(4.0999998))*((-t)))+(((t)*(40.42596))+(118.91168))
Generation 2458: Fitness 1874.318517879279: (((t)*(4.0999998))*((-t)))+(((t)*(40.42796))+(118.85768))
Generation 2464: Fitness 1874.318511117421: (((t)*(4.0999998))*((-t)))+(((t)*(40.42796))+(118.85788))
Generation 2473: Fitness 1874.318063663577: (((t)*(4.0999998))*((-t)))+(((t)*(40.42896))+(118.85768))
Generation 2477: Fitness 1849.582938502292: (((t)*(4.4999998))*((-t)))+(((t)*(44.42496))+(118.91468))
Generation 2481: Fitness 1873.462234709762: (((t)*(4.1053))*((-t)))+(((t)*(40.42596))+(118.91168))
Generation 2488: Fitness 1873.460886852012: (((t)*(4.1053))*((-t)))+(((t)*(40.42596))+(118.91468))
Generation 2492: Fitness 1873.457730119462: (((t)*(4.1053))*((-t)))+(((t)*(40.42696))+(118.91168))
Generation 2494: Fitness 1872.282398404166: (((t)*(4.1253))*((-t)))+(((t)*(40.42596))+(118.91168))
Generation 2500: Fitness 1871.539930080213: (((t)*(4.1653))*((-t)))+(((t)*(41.42596))+(118.91169))
Generation 2508: Fitness 1870.601895874166: (((t)*(4.1253))*((-t)))+(((t)*(40.52596))+(118.91168))
Generation 2510: Fitness 1869.74835477595: (((t)*(4.1353))*((-t)))+(((t)*(40.92596))+(118.91468))
Generation 2515: Fitness 1868.218732926046: (((t)*(4.1653))*((-t)))+(((t)*(41.42596))+(118.26169))
Generation 2516: Fitness 1865.374833820771: (((t)*(4.1553))*((-t)))+(((t)*(40.82596))+(118.91168))
Generation 2517: Fitness 1863.11394680595: (((t)*(4.1853))*((-t)))+(((t)*(41.42596))+(118.91169))
Generation 2521: Fitness 1859.837795759485: (((t)*(4.1953))*((-t)))+(((t)*(41.42596))+(118.81169))
Generation 2529: Fitness 1858.911622126152: (((t)*(4.1953))*((-t)))+(((t)*(41.42596))+(118.41169))
Generation 2533: Fitness 1858.406927592818: (((t)*(4.1953))*((-t)))+(((t)*(41.42596))+(116.81169))
Generation 2534: Fitness 1858.305448492818: (((t)*(4.1953))*((-t)))+(((t)*(41.42596))+(118.01169))
Generation 2536: Fitness 1854.243635407351: (((t)*(4.5353))*((-t)))+(((t)*(44.82596))+(118.91468))
Generation 2539: Fitness 1850.656457937835: (((t)*(4.2913))*((-t)))+(((t)*(41.92596))+(118.91169))
Generation 2540: Fitness 1829.425638146964: (((t)*(4.6653))*((-t)))+(((t)*(46.12596))+(114.91169))
Generation 2545: Fitness 1825.650512407351: (((t)*(4.5353))*((-t)))+(((t)*(44.82596))+(114.91468))
Generation 2554: Fitness 1820.859039629071: (((t)*(4.6653))*((-t)))+(((t)*(46.12516))+(112.91169))
Generation 2557: Fitness 1820.849903313631: (((t)*(4.6653))*((-t)))+(((t)*(46.12596))+(112.91169))
Generation 2563: Fitness 1818.445749873751: (((t)*(4.5353))*((-t)))+(((t)*(45.92546))+(108.11468))
Generation 2565: Fitness 1816.726952973751: (((t)*(4.5353))*((-t)))+(((t)*(45.92546))+(106.91468))
Generation 2568: Fitness 1813.463850576961: (((t)*(4.5953))*((-t)))+(((t)*(45.92546))+(108.91468))
Generation 2572: Fitness 1812.790747878558: (((t)*(4.5753))*((-t)))+(((t)*(45.92546))+(108.91468))
Generation 2573: Fitness 1812.790733027695: (((t)*(4.5753))*((-t)))+(((t)*(45.92547))+(108.91468))
Generation 2577: Fitness 1812.75672836891: (((t)*(4.5753))*((-t)))+(((t)*(45.92546))+(108.81468))
Generation 2584: Fitness 1812.402605522628: (((t)*(4.5953))*((-t)))+(((t)*(45.99546))+(108.91468))
Generation 2592: Fitness 1812.29284467523: (((t)*(4.5853))*((-t)))+(((t)*(45.98547))+(108.91468))
Generation 2594: Fitness 1812.227114848277: (((t)*(4.5903))*((-t)))+(((t)*(45.99546))+(108.91468))
Generation 2599: Fitness 1812.22708511776: (((t)*(4.5903))*((-t)))+(((t)*(45.99546))+(108.91488))
Generation 2609: Fitness 1812.224084741757: (((t)*(4.5903))*((-t)))+(((t)*(45.99586))+(108.91468))
Generation 2611: Fitness 1812.22023366756: (((t)*(4.5853))*((-t)))+(((t)*(45.99946))+(108.61468))
Generation 2612: Fitness 1812.219291163587: (((t)*(4.5873))*((-t)))+(((t)*(45.99546))+(108.91468))
Generation 2615: Fitness 1812.197495354651: (((t)*(4.5903))*((-t)))+(((t)*(45.99946))+(108.91469))
Generation 2621: Fitness 1812.19747615656: (((t)*(4.5903))*((-t)))+(((t)*(45.99946))+(108.91488))
Generation 2625: Fitness 1812.193900710677: (((t)*(4.5903))*((-t)))+(((t)*(45.99996))+(108.91468))
Generation 2628: Fitness 1812.193899757751: (((t)*(4.5903))*((-t)))+(((t)*(45.99996))+(108.91469))
Generation 2635: Fitness 1812.193890239493: (((t)*(4.5903))*((-t)))+(((t)*(45.99996))+(108.91479))
Generation 2636: Fitness 1812.191941280251: (((t)*(4.5903))*((-t)))+(((t)*(45.99996))+(108.94469))
Generation 2638: Fitness 1812.191636560994: (((t)*(4.5903))*((-t)))+(((t)*(45.99996))+(108.96488))
Generation 2640: Fitness 1812.191635581511: (((t)*(4.5903))*((-t)))+(((t)*(45.99996))+(108.96468))
Generation 2646: Fitness 1812.191169571366: (((t)*(4.5902))*((-t)))+(((t)*(45.99996))+(108.94469))
Generation 2650: Fitness 1812.191011548866: (((t)*(4.5902))*((-t)))+(((t)*(45.99996))+(108.95469))
Generation 2651: Fitness 1812.191169571366: (((t)*(4.5902))*((-t)))+(((t)*(45.99996))+(108.94469))
Generation 2653: Fitness 1812.190479088081: (((t)*(4.5901))*((-t)))+(((t)*(45.99996))+(108.94469))
Generation 2656: Fitness 1812.190415832233: (((t)*(4.5901))*((-t)))+(((t)*(45.99996))+(108.95468))
Generation 2657: Fitness 1812.190346295822: (((t)*(4.5901))*((-t)))+(((t)*(45.99998))+(108.94469))
Generation 2667: Fitness 1812.190279913913: (((t)*(4.5901))*((-t)))+(((t)*(45.99999))+(108.94469))
Generation 2675: Fitness 1812.190220224496: (((t)*(4.5901))*((-t)))+(((t)*(45.99999))+(108.95468))
Generation 2678: Fitness 1812.190219832987: (((t)*(4.5901))*((-t)))+(((t)*(45.99999))+(108.95458))
Generation 2685: Fitness 1812.19021667377: (((t)*(4.5901))*((-t)))+(((t)*(45.99999))+(108.95198))
Generation 2689: Fitness 1812.190216194329: (((t)*(4.5901))*((-t)))+(((t)*(45.99999))+(108.95268))
Generation 2700: Fitness 1812.190216194278: (((t)*(4.5901))*((-t)))+(((t)*(45.99999))+(108.95267))
Generation 2722: Fitness 1812.188561737904: (((t)*(4.5891))*((-t)))+(((t)*(45.99999))+(108.95268))
Generation 2724: Fitness 1812.190216194278: (((t)*(4.5901))*((-t)))+(((t)*(45.99999))+(108.95267))
Generation 2726: Fitness 1812.190150771469: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95267))
Generation 2743: Fitness 1812.190150769588: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95265))
Generation 2744: Fitness 1812.190150768506: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95263))
Generation 2745: Fitness 1812.190150769588: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95265))
Generation 2746: Fitness 1812.190150768506: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95263))
Generation 2747: Fitness 1812.190150768265: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95262))
Generation 2768: Fitness 1812.190150768225: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95261))
Generation 2784: Fitness 1811.981528893225: (((t)*(4.5901))*((-t)))+(((t)*(46.05))+(108.95261))
Generation 2787: Fitness 1812.190150768225: (((t)*(4.5901))*((-t)))+(((t)*(46))+(108.95261))
```

```
Generation 2790: Fitness 1809.267484173708: (((t)*(4.6901))*((-t)))+(((t)*(47))+(108.95261))
Generation 2801: Fitness 1809.258342651524: (((t)*(4.6904))*((-t)))+(((t)*(47))+(108.95261))
Generation 2805: Fitness 1807.847823423708: (((t)*(4.6901))*((-t)))+(((t)*(47))+(107.95261))
Generation 2812: Fitness 1807.622819336159: (((t)*(4.6801))*((-t)))+(((t)*(47))+(107.95261))
Generation 2814: Fitness 1807.255125252826: (((t)*(4.6801))*((-t)))+(((t)*(47))+(106.95261))
Generation 2818: Fitness 1806.894343831278: (((t)*(4.6701))*((-t)))+(((t)*(47))+(106.95261))
Generation 2828: Fitness 1806.894029103861: (((t)*(4.6701))*((-t)))+(((t)*(47))+(106.95161))
Generation 2829: Fitness 1806.872771089611: (((t)*(4.6701))*((-t)))+(((t)*(47))+(106.85261))
Generation 2835: Fitness 1806.871198347945: (((t)*(4.6701))*((-t)))+(((t)*(47))+(106.75261))
Generation 2838: Fitness 1806.578306280304: (((t)*(4.6921))*((-t)))+(((t)*(47.14))+(107.02261))
Generation 2840: Fitness 1805.96417009434: (((t)*(4.7801))*((-t)))+(((t)*(48))+(106.95265))
Generation 2841: Fitness 1806.578306280304: (((t)*(4.6921))*((-t)))+(((t)*(47.14))+(107.02261))
Generation 2848: Fitness 1806.330187313637: (((t)*(4.6921))*((-t)))+(((t)*(47.18))+(107.02261))
Generation 2849: Fitness 1806.578306280304: (((t)*(4.6921))*((-t)))+(((t)*(47.14))+(107.02261))
Generation 2857: Fitness 1806.502056038637: (((t)*(4.6921))*((-t)))+(((t)*(47.15))+(107.02261))
Generation 2858: Fitness 1806.435286130304: (((t)*(4.6921))*((-t)))+(((t)*(47.16))+(107.02261))
Generation 2860: Fitness 1806.33667176992: (((t)*(4.6925))*((-t)))+(((t)*(47.18))+(107.02261))
Generation 2861: Fitness 1806.330187313637: (((t)*(4.6921))*((-t)))+(((t)*(47.18))+(107.02261))
Generation 2863: Fitness 1806.242418863638: (((t)*(4.6921))*((-t)))+(((t)*(47.24))+(107.02261))
Generation 2868: Fitness 1806.233346105304: (((t)*(4.6921))*((-t)))+(((t)*(47.23))+(107.02261))
Generation 2873: Fitness 1806.016364780304: (((t)*(4.6921))*((-t)))+(((t)*(47.24))+(106.02261))
Generation 2878: Fitness 1805.824951296971: (((t)*(4.6921))*((-t)))+(((t)*(47.26))+(106.02261))
Generation 2883: Fitness 1805.674538606138: (((t)*(4.6921))*((-t)))+(((t)*(47.28))+(106.01261))
Generation 2884: Fitness 1805.671488951562: (((t)*(4.6921))*((-t)))+(((t)*(47.28))+(106.02251))
Generation 2886: Fitness 1805.608933571971: (((t)*(4.6921))*((-t)))+(((t)*(47.29))+(106.02261))
Generation 2890: Fitness 1805.438510696971: (((t)*(4.6921))*((-t)))+(((t)*(47.34))+(106.02261))
Generation 2891: Fitness 1805.435895745975: (((t)*(4.6922))*((-t)))+(((t)*(47.34))+(106.02261))
Generation 2896: Fitness 1805.217716893598: (((t)*(4.6991))*((-t)))+(((t)*(47.44))+(106.01261))
Generation 2908: Fitness 1805.217532447448: (((t)*(4.6991))*((-t)))+(((t)*(47.44))+(106.01241))
Generation 2912: Fitness 1805.207136207863: (((t)*(4.6993))*((-t)))+(((t)*(47.44))+(106.01261))
Generation 2915: Fitness 1804.932831576098: (((t)*(4.6991))*((-t)))+(((t)*(47.49))+(105.02261))
Generation 2918: Fitness 1804.932822325113: (((t)*(4.6991))*((-t)))+(((t)*(47.49))+(105.02263))
Generation 2923: Fitness 1804.880316475113: (((t)*(4.6991))*((-t)))+(((t)*(47.49))+(105.22263))
Generation 2932: Fitness 1804.876217736832: (((t)*(4.6993))*((-t)))+(((t)*(47.49))+(105.22263))
Generation 2936: Fitness 1804.87244390095: (((t)*(4.6995))*((-t)))+(((t)*(47.49))+(105.22263))
Generation 2938: Fitness 1804.867392339128: (((t)*(4.6998))*((-t)))+(((t)*(47.49))+(105.22263))
Generation 2939: Fitness 1804.863269928294: (((t)*(4.6998))*((-t)))+(((t)*(47.49))+(105.29263))
Generation 2944: Fitness 1804.835526140113: (((t)*(4.6991))*((-t)))+(((t)*(47.52))+(105.02263))
Generation 2953: Fitness 1804.818031610794: (((t)*(4.6998))*((-t)))+(((t)*(47.52))+(105.02263))
Generation 2958: Fitness 1804.815147807053: (((t)*(4.6999))*((-t)))+(((t)*(47.52))+(105.02663))
Generation 2960: Fitness 1804.813818364969: (((t)*(4.6997))*((-t)))+(((t)*(47.52))+(105.09263))
Generation 2961: Fitness 1804.802891732469: (((t)*(4.6997))*((-t)))+(((t)*(47.56))+(105.02263))
Generation 2962: Fitness 1804.813818364969: (((t)*(4.6997))*((-t)))+(((t)*(47.52))+(105.09263))
Generation 2964: Fitness 1804.810202418053: (((t)*(4.6999))*((-t)))+(((t)*(47.52))+(105.06263))
Generation 2966: Fitness 1804.792859107053: (((t)*(4.6999))*((-t)))+(((t)*(47.56))+(105.02663))
Generation 2974: Fitness 1804.784800278053: (((t)*(4.6999))*((-t)))+(((t)*(47.54))+(105.02263))
Generation 2975: Fitness 1804.783494026101: (((t)*(4.6999))*((-t)))+(((t)*(47.55))+(105.02264))
Generation 2981: Fitness 1804.783317641637: (((t)*(4.6999))*((-t)))+(((t)*(47.55))+(105.02163))
Generation 2986: Fitness 1804.78247450122: (((t)*(4.6999))*((-t)))+(((t)*(47.55))+(105.01663))
Generation 2987: Fitness 1804.781007867579: (((t)*(4.7))*((-t)))+(((t)*(47.54))+(105.02263))
Generation 2996: Fitness 1804.780998478029: (((t)*(4.7))*((-t)))+(((t)*(47.54))+(105.02243))
Generation 2999: Fitness 1804.780984543704: (((t)*(4.7))*((-t)))+(((t)*(47.54))+(105.02213))
Generation 3000: Fitness 1804.780998478029: (((t)*(4.7))*((-t)))+(((t)*(47.54))+(105.02243))
Generation 3001: Fitness 1804.780993813254: (((t)*(4.7))*((-t)))+(((t)*(47.54))+(105.02233))
Generation 3002: Fitness 1804.780636390079: (((t)*(4.7))*((-t)))+(((t)*(47.54))+(105.01263))
Generation 3007: Fitness 1804.777288721746: (((t)*(4.7))*((-t)))+(((t)*(47.55))+(105.01263))
Generation 3008: Fitness 1804.777143573996: (((t)*(4.7))*((-t)))+(((t)*(47.55))+(105.01163))
Generation 3009: Fitness 1804.775764942171: (((t)*(4.7))*((-t)))+(((t)*(47.55))+(105.00133))
Generation 3014: Fitness 1804.775727967846: (((t)*(4.7))*((-t)))+(((t)*(47.55))+(105.00103))
Generation 3019: Fitness 1804.737850611148: (((t)*(4.7099))*((-t)))+(((t)*(47.55))+(105.00183))
Generation 3024: Fitness 1804.733805584898: (((t)*(4.7099))*((-t)))+(((t)*(47.55))+(105.00683))
Generation 3032: Fitness 1804.730738927044: (((t)*(4.7097))*((-t)))+(((t)*(47.55))+(105.00183))
Generation 3034: Fitness 1804.664631705641: (((t)*(4.7039))*((-t)))+(((t)*(47.55))+(105.00683))
Generation 3036: Fitness 1804.526224959898: (((t)*(4.7099))*((-t)))+(((t)*(47.58))+(105.00683))
Generation 3038: Fitness 1804.371776861148: (((t)*(4.7099))*((-t)))+(((t)*(47.65))+(105.00183))
Generation 3043: Fitness 1804.365722736148: (((t)*(4.7099))*((-t)))+(((t)*(47.64))+(105.00183))
Generation 3050: Fitness 1804.365697046673: (((t)*(4.7099))*((-t)))+(((t)*(47.64))+(105.00173))
Generation 3053: Fitness 1804.291951626043: (((t)*(4.7199))*((-t)))+(((t)*(47.65))+(105.00183))
Generation 3054: Fitness 1804.289116433127: (((t)*(4.7199))*((-t)))+(((t)*(47.65))+(105.00683))
Generation 3058: Fitness 1804.24474776771: (((t)*(4.7199))*((-t)))+(((t)*(47.65))+(105.10183))
Generation 3060: Fitness 1804.222520084377: (((t)*(4.7199))*((-t)))+(((t)*(47.66))+(105.00183))
Generation 3066: Fitness 1804.162568876043: (((t)*(4.7199))*((-t)))+(((t)*(47.67))+(105.00183))
Generation 3067: Fitness 1804.109275775969: (((t)*(4.7193))*((-t)))+(((t)*(47.68))+(105.00183))
Generation 3068: Fitness 1804.057114704302: (((t)*(4.7193))*((-t)))+(((t)*(47.75))+(105.00183))
Generation 3070: Fitness 1804.024251209377: (((t)*(4.7199))*((-t)))+(((t)*(47.75))+(105.00183))
Generation 3072: Fitness 1804.024238850548: (((t)*(4.7199))*((-t)))+(((t)*(47.75))+(105.00181))
Generation 3073: Fitness 1804.024251209377: (((t)*(4.7199))*((-t)))+(((t)*(47.75))+(105.00183))
Generation 3078: Fitness 1803.18627507166: (((t)*(4.8199))*((-t)))+(((t)*(48.65))+(105.00183))
Generation 3094: Fitness 1803.177221381827: (((t)*(4.8199))*((-t)))+(((t)*(48.64))+(105.00783))
Generation 3095: Fitness 1803.166813613327: (((t)*(4.8199))*((-t)))+(((t)*(48.64))+(105.00183))
Generation 3097: Fitness 1803.15633169666: (((t)*(4.8199))*((-t)))+(((t)*(48.62))+(105.00183))
Generation 3099: Fitness 1802.272277285131: (((t)*(4.8119))*((-t)))+(((t)*(48.65))+(103.00183))
Generation 3105: Fitness 1802.190370588572: (((t)*(4.8112))*((-t)))+(((t)*(48.65))+(103.00183))
Generation 3108: Fitness 1802.190304211927: (((t)*(4.8112))*((-t)))+(((t)*(48.65))+(103.00188))
```

```
Generation 3110: Fitness 1801.897856452752: (((t)*(4.8126))*((-t)))+(((t)*(48.65))+(104.00183))
Generation 3113: Fitness 1801.288038201797: (((t)*(4.8119))*((-t)))+(((t)*(48.75))+(103.00183))
Generation 3115: Fitness 1801.265600828593: (((t)*(4.8112))*((-t)))+(((t)*(48.75))+(103.00188))
Generation 3119: Fitness 1801.16071392509: (((t)*(4.8113))*((-t)))+(((t)*(48.79))+(103.00183))
Generation 3127: Fitness 1801.159791901797: (((t)*(4.8119))*((-t)))+(((t)*(48.79))+(103.00183))
Generation 3130: Fitness 1801.070067347522: (((t)*(4.8312))*((-t)))+(((t)*(48.95))+(103.00183))
Generation 3135: Fitness 1801.070032722481: (((t)*(4.8312))*((-t)))+(((t)*(48.95))+(103.00173))
Generation 3141: Fitness 1801.069928967356: (((t)*(4.8312))*((-t)))+(((t)*(48.95))+(103.00143))
Generation 3142: Fitness 1801.025498774427: (((t)*(4.8112))*((-t)))+(((t)*(48.85))+(102.59188))
Generation 3147: Fitness 1801.023849316928: (((t)*(4.8113))*((-t)))+(((t)*(48.85))+(102.59188))
Generation 3154: Fitness 1801.019388298032: (((t)*(4.8116))*((-t)))+(((t)*(48.85))+(102.59188))
Generation 3156: Fitness 1801.014122578593: (((t)*(4.8112))*((-t)))+(((t)*(48.85))+(102.52188))
Generation 3159: Fitness 1801.012247066093: (((t)*(4.8112))*((-t)))+(((t)*(48.85))+(102.49188))
Generation 3163: Fitness 1801.007942488442: (((t)*(4.8132))*((-t)))+(((t)*(48.85))+(102.59188))
Generation 3164: Fitness 1800.99224534776: (((t)*(4.8112))*((-t)))+(((t)*(48.87))+(102.39188))
Generation 3170: Fitness 1800.992196475802: (((t)*(4.8112))*((-t)))+(((t)*(48.87))+(102.39118))
Generation 3173: Fitness 1800.984747466767: (((t)*(4.8118))*((-t)))+(((t)*(48.87))+(102.39188))
Generation 3174: Fitness 1800.984742172734: (((t)*(4.8118))*((-t)))+(((t)*(48.87))+(102.39148))
Generation 3180: Fitness 1800.981390881812: (((t)*(4.8122))*((-t)))+(((t)*(48.87))+(102.39118))
Generation 3181: Fitness 1800.978623995108: (((t)*(4.8132))*((-t)))+(((t)*(48.87))+(102.39188))
Generation 3183: Fitness 1800.972436729867: (((t)*(4.8149))*((-t)))+(((t)*(48.89))+(102.59188))
Generation 3184: Fitness 1800.97862399508: (((t)*(4.8132))*((-t)))+(((t)*(48.87))+(102.39188))
Generation 3188: Fitness 1800.944188882106: (((t)*(4.8142))*((-t)))+(((t)*(48.89))+(102.39188))
Generation 3192: Fitness 1800.944187679252: (((t)*(4.8142))*((-t)))+(((t)*(48.89))+(102.39183))
Generation 3193: Fitness 1800.944172497148: (((t)*(4.8142))*((-t)))+(((t)*(48.89))+(102.39118))
Generation 3195: Fitness 1800.894809738362: (((t)*(4.8182))*((-t)))+(((t)*(48.97))+(102.33188))
Generation 3196: Fitness 1800.942839582038: (((t)*(4.8144))*((-t)))+(((t)*(48.89))+(102.39188))
Generation 3197: Fitness 1800.941221724725: (((t)*(4.8172))*((-t)))+(((t)*(48.89))+(102.53188))
Generation 3199: Fitness 1800.939856539414: (((t)*(4.8162))*((-t)))+(((t)*(48.89))+(102.51188))
Generation 3203: Fitness 1800.939189895081: (((t)*(4.8162))*((-t)))+(((t)*(48.89))+(102.50148))
Generation 3210: Fitness 1800.929032825768: (((t)*(4.8148))*((-t)))+(((t)*(48.91))+(102.39188))
Generation 3212: Fitness 1800.927105596533: (((t)*(4.8149))*((-t)))+(((t)*(48.91))+(102.39188))
Generation 3216: Fitness 1800.905197589725: (((t)*(4.8172))*((-t)))+(((t)*(48.91))+(102.39188))
Generation 3228: Fitness 1800.861897241993: (((t)*(4.8192))*((-t)))+(((t)*(48.94))+(102.39188))
Generation 3238: Fitness 1800.857693625327: (((t)*(4.8192))*((-t)))+(((t)*(48.94))+(102.35188))
Generation 3240: Fitness 1800.85714272116: (((t)*(4.8192))*((-t)))+(((t)*(48.94))+(102.34188))
Generation 3241: Fitness 1800.857693625327: (((t)*(4.8192))*((-t)))+(((t)*(48.94))+(102.35188))
Generation 3242: Fitness 1800.857286999731: (((t)*(4.8199))*((-t)))+(((t)*(48.94))+(102.39188))
Generation 3244: Fitness 1800.85714272116: (((t)*(4.8192))*((-t)))+(((t)*(48.94))+(102.34188))
Generation 3245: Fitness 1800.85669000866: (((t)*(4.8192))*((-t)))+(((t)*(48.94))+(102.31188))
Generation 3246: Fitness 1800.85714272116: (((t)*(4.8192))*((-t)))+(((t)*(48.94))+(102.34188))
Generation 3247: Fitness 1800.856112438064: (((t)*(4.8199))*((-t)))+(((t)*(48.94))+(102.37188))
Generation 3250: Fitness 1800.841518462481: (((t)*(4.8192))*((-t)))+(((t)*(48.96))+(102.29177))
Generation 3252: Fitness 1800.849761101397: (((t)*(4.8199))*((-t)))+(((t)*(48.97))+(102.35188))
Generation 3253: Fitness 1800.828318424133: (((t)*(4.8272))*((-t)))+(((t)*(48.98))+(102.39188))
Generation 3256: Fitness 1800.82014861366: (((t)*(4.8192))*((-t)))+(((t)*(48.98))+(102.05188))
Generation 3263: Fitness 1800.819659517826: (((t)*(4.8192))*((-t)))+(((t)*(48.98))+(102.06188))
Generation 3266: Fitness 1800.818615484731: (((t)*(4.8199))*((-t)))+(((t)*(48.97))+(102.15188))
Generation 3267: Fitness 1800.810814580556: (((t)*(4.8262))*((-t)))+(((t)*(48.98))+(102.39188))
Generation 3269: Fitness 1800.807038409783: (((t)*(4.8232))*((-t)))+(((t)*(48.98))+(102.39188))
Generation 3273: Fitness 1800.800174573381: (((t)*(4.8242))*((-t)))+(((t)*(48.98))+(102.39188))
Generation 3275: Fitness 1800.788494742222: (((t)*(4.8262))*((-t)))+(((t)*(48.99))+(102.39188))
Generation 3279: Fitness 1800.787365400556: (((t)*(4.8262))*((-t)))+(((t)*(48.99))+(102.37188))
Generation 3284: Fitness 1800.787361763847: (((t)*(4.8262))*((-t)))+(((t)*(48.99))+(102.37178))
Generation 3291: Fitness 1800.787036058889: (((t)*(4.8262))*((-t)))+(((t)*(48.99))+(102.35188))
Generation 3292: Fitness 1800.786989224873: (((t)*(4.8256))*((-t)))+(((t)*(48.99))+(102.39188))
Generation 3293: Fitness 1800.787036058889: (((t)*(4.8262))*((-t)))+(((t)*(48.99))+(102.35188))
Generation 3298: Fitness 1800.784584650305: (((t)*(4.8252))*((-t)))+(((t)*(48.99))+(102.37188))
Generation 3300: Fitness 1800.780521129472: (((t)*(4.8252))*((-t)))+(((t)*(48.99))+(102.32188))
Generation 3309: Fitness 1800.780308425305: (((t)*(4.8252))*((-t)))+(((t)*(48.99))+(102.31188))
Generation 3311: Fitness 1800.78030513418: (((t)*(4.8252))*((-t)))+(((t)*(48.99))+(102.31158))
Generation 3312: Fitness 1800.780295721138: (((t)*(4.8252))*((-t)))+(((t)*(48.99))+(102.30188))
Generation 3317: Fitness 1800.777311352922: (((t)*(4.8242))*((-t)))+(((t)*(48.99))+(102.27158))
Generation 3322: Fitness 1800.777308815547: (((t)*(4.8242))*((-t)))+(((t)*(48.99))+(102.27148))
Generation 3328: Fitness 1800.777157750172: (((t)*(4.8242))*((-t)))+(((t)*(48.99))+(102.26178))
Generation 3331: Fitness 1800.777156615422: (((t)*(4.8242))*((-t)))+(((t)*(48.99))+(102.26158))
Generation 3336: Fitness 1800.776918139909: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24188))
Generation 3338: Fitness 1800.776925984669: (((t)*(4.8239))*((-t)))+(((t)*(48.99))+(102.24188))
Generation 3340: Fitness 1800.776918139909: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24188))
Generation 3351: Fitness 1800.776917380893: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24168))
Generation 3355: Fitness 1800.776917031384: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24158))
Generation 3356: Fitness 1800.776915833351: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24118))
Generation 3367: Fitness 1800.776915681246: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24112))
Generation 3370: Fitness 1800.776914307776: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24028))
Generation 3373: Fitness 1800.776914238268: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.24018))
Generation 3374: Fitness 1800.77691415024: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.23986))
Generation 3395: Fitness 1800.776914149743: (((t)*(4.8238))*((-t)))+(((t)*(48.99))+(102.23988))
Generation 3806: Fitness 1800.766378484743: (((t)*(4.8238))*((-t)))+(((t)*(49))+(102.23988))
Generation 3816: Fitness 1800.76180509103: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.23988))
Generation 3824: Fitness 1800.76180436615: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.23985))
Generation 3825: Fitness 1800.761795574303: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.23948))
Generation 3826: Fitness 1800.76170913225 3: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.23488))
Generation 3827: Fitness 1800.761663173503: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.22988))
Generation 3832: Fitness 1800.761662764328: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.22978))
```

```
Generation 3837: Fitness 1800.761662005978: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.22958))
Generation 3840: Fitness 1800.761659135878: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.22838))
Generation 3842: Fitness 1800.761659598253: (((t)*(4.8248))*((-t)))+(((t)*(49))+(102.22688))
Generation 3845: Fitness 1800.76162347351: (((t)*(4.8247))*((-t)))+(((t)*(49))+(102.21988))
Generation 3849: Fitness 1800.76161348976: (((t)*(4.8247))*((-t)))+(((t)*(49))+(102.22288))
Generation 3854: Fitness 1800.761613466969: (((t)*(4.8247))*((-t)))+(((t)*(49))+(102.22298))
Generation 3863: Fitness 1800.761606916618: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21988))
Generation 3874: Fitness 1800.761604764201: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21888))
Generation 3879: Fitness 1800.761604611785: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21788))
Generation 3880: Fitness 1800.761604764201: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21888))
Generation 3882: Fitness 1800.761604611785: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21788))
Generation 3888: Fitness 1800.76160444751: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21818))
Generation 3890: Fitness 1800.761604437993: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21838))
Generation 3891: Fitness 1800.761604432375: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21829))
Generation 3911: Fitness 1800.761604432224: (((t)*(4.8246))*((-t)))+(((t)*(49))+(102.21831))
Generation 4272: Fitness 1800.67614379469: (((t)*(4.8546))*((-t)))+(((t)*(49.237))+(102.21831))
Generation 4277: Fitness 1800.67614260459: (((t)*(4.8546))*((-t)))+(((t)*(49.237))+(102.21836))
Generation 4279: Fitness 1800.660115171554: (((t)*(4.8446))*((-t)))+(((t)*(49.237))+(102.21831))
Generation 4281: Fitness 1800.67614260459: (((t)*(4.8546))*((-t)))+(((t)*(49.237))+(102.21836))
Generation 4282: Fitness 1800.567779416469: (((t)*(4.8546))*((-t)))+(((t)*(49.287))+(102.21831))
Generation 4293: Fitness 1800.567665213985: (((t)*(4.8546))*((-t)))+(((t)*(49.287))+(102.21811))
Generation 4296: Fitness 1800.567494924257: (((t)*(4.8536))*((-t)))+(((t)*(49.287))+(102.21131))
Generation 4297: Fitness 1800.520583972319: (((t)*(4.8546))*((-t)))+(((t)*(49.287))+(102.11811))
Generation 4300: Fitness 1800.563829929552: (((t)*(4.8546))*((-t)))+(((t)*(49.287))+(102.21131))
Generation 4304: Fitness 1800.520658174802: (((t)*(4.8546))*((-t)))+(((t)*(49.287))+(102.11831))
Generation 4307: Fitness 1800.519718372302: (((t)*(4.8546))*((-t)))+(((t)*(49.288))+(102.11831))
Generation 4310: Fitness 1800.518108687885: (((t)*(4.8546))*((-t)))+(((t)*(49.287))+(102.11131))
Generation 4311: Fitness 1800.517085585385: (((t)*(4.8546))*((-t)))+(((t)*(49.288))+(102.11131))
Generation 4315: Fitness 1800.515526299802: (((t)*(4.8546))*((-t)))+(((t)*(49.297))+(102.11831))
Generation 4318: Fitness 1800.515506459507: (((t)*(4.8536))*((-t)))+(((t)*(49.287))+(102.11831))
Generation 4319: Fitness 1800.515526299802: (((t)*(4.8546))*((-t)))+(((t)*(49.297))+(102.11831))
Generation 4320: Fitness 1800.515511217498: (((t)*(4.8543))*((-t)))+(((t)*(49.287))+(102.11131))
Generation 4321: Fitness 1800.515276788122: (((t)*(4.8526))*((-t)))+(((t)*(49.288))+(102.11131))
Generation 4322: Fitness 1800.51367213788: (((t)*(4.8536))*((-t)))+(((t)*(49.287))+(102.11434))
Generation 4326: Fitness 1800.512068743719: (((t)*(4.8546))*((-t)))+(((t)*(49.298))+(102.11131))
Generation 4331: Fitness 1800.490174343719: (((t)*(4.8546))*((-t)))+(((t)*(49.288))+(102.01131))
Generation 4332: Fitness 1800.472395395872: (((t)*(4.8526))*((-t)))+(((t)*(49.287))+(102.01831))
Generation 4336: Fitness 1800.469333175972: (((t)*(4.8526))*((-t)))+(((t)*(49.288))+(102.01111))
Generation 4337: Fitness 1800.467308090091: (((t)*(4.8536))*((-t)))+(((t)*(49.298))+(102.01131))
Generation 4343: Fitness 1800.467107165701: (((t)*(4.8533))*((-t)))+(((t)*(49.298))+(102.01131))
Generation 4355: Fitness 1800.467092914897: (((t)*(4.8534))*((-t)))+(((t)*(49.298))+(102.01131))
Generation 4358: Fitness 1800.466987484257: (((t)*(4.8536))*((-t)))+(((t)*(49.299))+(102.01131))
Generation 4367: Fitness 1800.466942241064: (((t)*(4.8534))*((-t)))+(((t)*(49.299))+(102.01131))
Generation 4371: Fitness 1800.466908561107: (((t)*(4.8536))*((-t)))+(((t)*(49.299))+(102.01111))
Generation 4379: Fitness 1800.466843430644: (((t)*(4.8535))*((-t)))+(((t)*(49.299))+(102.01111))
Generation 4391: Fitness 1800.463064403607: (((t)*(4.8536))*((-t)))+(((t)*(49.299))+(102.00111))
Generation 4393: Fitness 1800.462825761614: (((t)*(4.8534))*((-t)))+(((t)*(49.299))+(102.00111))
Generation 4398: Fitness 1800.462690987857: (((t)*(4.8536))*((-t)))+(((t)*(49.299))+(102.00011))
Generation 4401: Fitness 1800.462825761614: (((t)*(4.8534))*((-t)))+(((t)*(49.299))+(102.00111))
Generation 4402: Fitness 1800.436453822311: (((t)*(4.8535))*((-t)))+(((t)*(49.299))+(101.91111))
Generation 4403: Fitness 1800.462786433972: (((t)*(4.8534))*((-t)))+(((t)*(49.299))+(102.00101))
Generation 4449: Fitness 1800.427101241725: (((t)*(4.8534))*((-t)))+(((t)*(49.299))+(101.75))
Generation 4459: Fitness 1800.394972323713: (((t)*(4.8514))*((-t)))+(((t)*(49.299))+(101.75))
Generation 4460: Fitness 1800.331042467306: (((t)*(4.8534))*((-t)))+(((t)*(49.399))+(101.00101))
Generation 4461: Fitness 1800.391091704696: (((t)*(4.8504))*((-t)))+(((t)*(49.299))+(101.75))
Generation 4475: Fitness 1800.38353506303: (((t)*(4.8504))*((-t)))+(((t)*(49.299))+(101.65))
Generation 4485: Fitness 1800.383390727196: (((t)*(4.8504))*((-t)))+(((t)*(49.299))+(101.66))
Generation 4490: Fitness 1800.381837579759: (((t)*(4.8502))*((-t)))+(((t)*(49.299))+(101.65))
Generation 4506: Fitness 1800.380699876388: (((t)*(4.85))*((-t)))+(((t)*(49.299))+(101.66))
Generation 4512: Fitness 1800.345195788092: (((t)*(4.8502))*((-t)))+(((t)*(49.399))+(101.15))
Generation 4516: Fitness 1800.380430121388: (((t)*(4.85))*((-t)))+(((t)*(49.299))+(101.64))
Generation 4519: Fitness 1800.320805452676: (((t)*(4.8592))*((-t)))+(((t)*(49.399))+(101.65))
Generation 4521: Fitness 1800.345426172075: (((t)*(4.8572))*((-t)))+(((t)*(49.399))+(101.65))
Generation 4523: Fitness 1800.338824576524: (((t)*(4.8501))*((-t)))+(((t)*(49.399))+(101.05))
Generation 4526: Fitness 1800.320678932967: (((t)*(4.8602))*((-t)))+(((t)*(49.399))+(101.65))
Generation 4530: Fitness 1800.31262750522: (((t)*(4.8521))*((-t)))+(((t)*(49.399))+(101.05))
Generation 4536: Fitness 1800.311712809558: (((t)*(4.8531))*((-t)))+(((t)*(49.399))+(101.05))
Generation 4541: Fitness 1800.31115483739: (((t)*(4.8526))*((-t)))+(((t)*(49.399))+(101.05))
Generation 4543: Fitness 1800.297064672075: (((t)*(4.8572))*((-t)))+(((t)*(49.399))+(101.25))
Generation 4544: Fitness 1800.281245422075: (((t)*(4.8572))*((-t)))+(((t)*(49.399))+(101.45))
Generation 4554: Fitness 1800.280136384575: (((t)*(4.8572))*((-t)))+(((t)*(49.399))+(101.44))
Generation 4558: Fitness 1800.279155047075: (((t)*(4.8572))*((-t)))+(((t)*(49.399))+(101.35))
Generation 4560: Fitness 1800.27833296478: (((t)*(4.8552))*((-t)))+(((t)*(49.399))+(101.25))
Generation 4568: Fitness 1800.276353381447: (((t)*(4.8552))*((-t)))+(((t)*(49.399))+(101.29))
Generation 4571: Fitness 1800.275208246764: (((t)*(4.8562))*((-t)))+(((t)*(49.399))+(101.35))
Generation 4573: Fitness 1800.276353381447: (((t)*(4.8552))*((-t)))+(((t)*(49.399))+(101.29))
Generation 4578: Fitness 1800.275208246764: (((t)*(4.8562))*((-t)))+(((t)*(49.399))+(101.35))
Generation 4595: Fitness 1800.275151175931: (((t)*(4.8562))*((-t)))+(((t)*(49.399))+(101.34))
Generation 4628: Fitness 1800.275108433366: (((t)*(4.8561))*((-t)))+(((t)*(49.399))+(101.34))
Generation 4630: Fitness 1800.274300325931: (((t)*(4.8562))*((-t)))+(((t)*(49.419))+(101.34))
Generation 4631: Fitness 1800.275108433366: (((t)*(4.8561))*((-t)))+(((t)*(49.399))+(101.34))
Generation 4695: Fitness 1800.215245092848: (((t)*(4.8661))*((-t)))+(((t)*(49.499))+(101.34))
Generation 4698: Fitness 1800.275108433366: (((t)*(4.8561))*((-t)))+(((t)*(49.399))+(101.34))
Generation 4944: Fitness 1800.215245092848: (((t)*(4.8661))*((-t)))+(((t)*(49.499))+(101.34))
```

```
Generation 4947: Fitness 1800.205814276181: (((t)*(4.8661))*((-t)))+(((t)*(49.499))+(101.14))
Generation 4955: Fitness 1800.200671980348: (((t)*(4.8661))*((-t)))+(((t)*(49.499))+(101.19))
Generation 4963: Fitness 1800.200529684514: (((t)*(4.8661))*((-t)))+(((t)*(49.499))+(101.24))
Generation 4964: Fitness 1800.191576267671: (((t)*(4.8641))*((-t)))+(((t)*(49.499))+(101.14))
Generation 4975: Fitness 1800.191308660171: (((t)*(4.8641))*((-t)))+(((t)*(49.499))+(101.13))
Generation 4978: Fitness 1800.191241052671: (((t)*(4.8641))*((-t)))+(((t)*(49.499))+(101.12))
Generation 5006: Fitness 1800.191225001686: (((t)*(4.8639))*((-t)))+(((t)*(49.499))+(101.12))
Generation 5048: Fitness 1800.191167787519: (((t)*(4.8639))*((-t)))+(((t)*(49.499))+(101.11))
Generation 5055: Fitness 1800.189258923352: (((t)*(4.8639))*((-t)))+(((t)*(49.509))+(101.12))
Generation 5062: Fitness 1800.159502198109: (((t)*(4.8839))*((-t)))+(((t)*(49.699))+(101.11))
Generation 5063: Fitness 1800.189258923352: (((t)*(4.8639))*((-t)))+(((t)*(49.509))+(101.12))
Generation 5064: Fitness 1800.164398745609: (((t)*(4.8839))*((-t)))+(((t)*(49.699))+(101.12))
Generation 5068: Fitness 1800.189258923352: (((t)*(4.8639))*((-t)))+(((t)*(49.509))+(101.12))
Generation 5073: Fitness 1800.188893712352: (((t)*(4.8639))*((-t)))+(((t)*(49.507))+(101.12))
Generation 5074: Fitness 1800.187046841198: (((t)*(4.8659))*((-t)))+(((t)*(49.509))+(101.12))
Generation 5076: Fitness 1800.184091602278: (((t)*(4.8649))*((-t)))+(((t)*(49.509))+(101.12))
Generation 5084: Fitness 1800.177569284531: (((t)*(4.8659))*((-t)))+(((t)*(49.529))+(101.12))
Generation 5090: Fitness 1800.177567896198: (((t)*(4.8659))*((-t)))+(((t)*(49.519))+(101.12))
Generation 5096: Fitness 1800.165201185814: (((t)*(4.8739))*((-t)))+(((t)*(49.576))+(101.12))
Generation 5112: Fitness 1800.164832734421: (((t)*(4.8731))*((-t)))+(((t)*(49.576))+(101.19))
Generation 5114: Fitness 1800.164608086114: (((t)*(4.8729))*((-t)))+(((t)*(49.576))+(101.19))
Generation 5115: Fitness 1800.163381219647: (((t)*(4.8739))*((-t)))+(((t)*(49.577))+(101.12))
Generation 5116: Fitness 1800.161045767355: (((t)*(4.8736))*((-t)))+(((t)*(49.576))+(101.12))
Generation 5117: Fitness 1800.156642367808: (((t)*(4.8732))*((-t)))+(((t)*(49.576))+(101.12))
Generation 5118: Fitness 1800.156543369195: (((t)*(4.8709))*((-t)))+(((t)*(49.576))+(101.12))
Generation 5119: Fitness 1800.147327207314: (((t)*(4.8739))*((-t)))+(((t)*(49.599))+(101.12))
Generation 5124: Fitness 1800.146814495814: (((t)*(4.8739))*((-t)))+(((t)*(49.596))+(101.12))
Generation 5127: Fitness 1800.144694614981: (((t)*(4.8739))*((-t)))+(((t)*(49.596))+(101.11))
Generation 5131: Fitness 1800.146814495814: (((t)*(4.8739))*((-t)))+(((t)*(49.596))+(101.12))
Generation 5133: Fitness 1800.145352746521: (((t)*(4.8736))*((-t)))+(((t)*(49.596))+(101.11))
Generation 5142: Fitness 1800.145235551355: (((t)*(4.8736))*((-t)))+(((t)*(49.595))+(101.11))
Generation 5143: Fitness 1800.145052143741: (((t)*(4.8737))*((-t)))+(((t)*(49.596))+(101.11))
Generation 5146: Fitness 1800.144832766561: (((t)*(4.8738))*((-t)))+(((t)*(49.596))+(101.11))
Generation 5148: Fitness 1800.144694614981: (((t)*(4.8739))*((-t)))+(((t)*(49.596))+(101.11))
Generation 5157: Fitness 1800.14465171548: (((t)*(4.8739))*((-t)))+(((t)*(49.597))+(101.11))
Generation 5168: Fitness 1800.132957268741: (((t)*(4.8737))*((-t)))+(((t)*(49.596))+(101.01))
Generation 5170: Fitness 1800.130291069421: (((t)*(4.8731))*((-t)))+(((t)*(49.596))+(101.01))
Generation 5177: Fitness 1800.130138115207: (((t)*(4.8727))*((-t)))+(((t)*(49.596))+(101.01))
Generation 5179: Fitness 1800.130054515361: (((t)*(4.8728))*((-t)))+(((t)*(49.596))+(101.01))
Generation 5183: Fitness 1800.129660243688: (((t)*(4.8736))*((-t)))+(((t)*(49.599))+(101.01))
Generation 5185: Fitness 1800.128969999628: (((t)*(4.8733))*((-t)))+(((t)*(49.599))+(101.01))
Generation 5188: Fitness 1800.128902369475: (((t)*(4.8732))*((-t)))+(((t)*(49.599))+(101.01))
Generation 5299: Fitness 1800.128497327808: (((t)*(4.8732))*((-t)))+(((t)*(49.599))+(100.91))
Generation 5311: Fitness 1800.126518840308: (((t)*(4.8732))*((-t)))+(((t)*(49.599))+(100.94))
Generation 5313: Fitness 1800.122808100937: (((t)*(4.8722))*((-t)))+(((t)*(49.599))+(100.91))
Generation 5329: Fitness 1800.122685232075: (((t)*(4.8719))*((-t)))+(((t)*(49.599))+(100.91))
Generation 5482: Fitness 1800.096424570503: (((t)*(4.8819))*((-t)))+(((t)*(49.699))+(100.91))
Generation 5484: Fitness 1800.122685232075: (((t)*(4.8719))*((-t)))+(((t)*(49.599))+(100.91))
Generation 5544: Fitness 1800.122539337075: (((t)*(4.8719))*((-t)))+(((t)*(49.599))+(100.89))
Generation 5560: Fitness 1800.122471225321: (((t)*(4.8718))*((-t)))+(((t)*(49.599))+(100.89))
Generation 5600: Fitness 1800.114921211455: (((t)*(4.8798))*((-t)))+(((t)*(49.699))+(100.89))
Generation 5604: Fitness 1800.091745270422: (((t)*(4.8818))*((-t)))+(((t)*(49.699))+(100.89))
Generation 5616: Fitness 1800.082308182922: (((t)*(4.8818))*((-t)))+(((t)*(49.699))+(100.84))
Generation 5619: Fitness 1800.076416686455: (((t)*(4.8798))*((-t)))+(((t)*(49.699))+(100.59))
Generation 5624: Fitness 1800.069251528122: (((t)*(4.8798))*((-t)))+(((t)*(49.699))+(100.69))
Generation 5643: Fitness 1800.069068043955: (((t)*(4.8798))*((-t)))+(((t)*(49.699))+(100.68))
Generation 5646: Fitness 1800.069010552306: (((t)*(4.8797))*((-t)))+(((t)*(49.699))+(100.67))
Generation 5654: Fitness 1800.069038319975: (((t)*(4.8795))*((-t)))+(((t)*(49.699))+(100.66))
Generation 5660: Fitness 1800.069010552306: (((t)*(4.8797))*((-t)))+(((t)*(49.699))+(100.67))
Generation 5662: Fitness 1800.069017770424: (((t)*(4.8796))*((-t)))+(((t)*(49.699))+(100.67))
Generation 5664: Fitness 1800.069010552306: (((t)*(4.8797))*((-t)))+(((t)*(49.699))+(100.67))
Generation 5722: Fitness 1800.053876103548: (((t)*(4.8897))*((-t)))+(((t)*(49.799))+(100.67))
Generation 5723: Fitness 1800.069010552306: (((t)*(4.8797))*((-t)))+(((t)*(49.699))+(100.67))
Generation 5848: Fitness 1800.068568368139: (((t)*(4.8797))*((-t)))+(((t)*(49.7))+(100.67))
Generation 5866: Fitness 1800.068557409622: (((t)*(4.8798))*((-t)))+(((t)*(49.7))+(100.67))
Generation 5949: Fitness 1800.05403265419: (((t)*(4.8898))*((-t)))+(((t)*(49.8))+(100.67))
Generation 5951: Fitness 1800.068557409622: (((t)*(4.8798))*((-t)))+(((t)*(49.7))+(100.67))
Generation 5956: Fitness 1800.05403265419: (((t)*(4.8898))*((-t)))+(((t)*(49.8))+(100.67))
Generation 5958: Fitness 1800.053728881266: (((t)*(4.8899))*((-t)))+(((t)*(49.8))+(100.67))
Generation 5970: Fitness 1800.049588352524: (((t)*(4.8898))*((-t)))+(((t)*(49.8))+(100.65))
Generation 5972: Fitness 1800.044592143766: (((t)*(4.8899))*((-t)))+(((t)*(49.8))+(100.62))
Generation 5976: Fitness 1800.036717898126: (((t)*(4.8891))*((-t)))+(((t)*(49.8))+(100.47))
Generation 5985: Fitness 1800.034134108097: (((t)*(4.8888))*((-t)))+(((t)*(49.8))+(100.47))
Generation 5988: Fitness 1800.030768920407: (((t)*(4.8879))*((-t)))+(((t)*(49.8))+(100.47))
Generation 5996: Fitness 1800.03064550624: (((t)*(4.8879))*((-t)))+(((t)*(49.8))+(100.46))
Generation 6005: Fitness 1800.030582921163: (((t)*(4.8878))*((-t)))+(((t)*(49.8))+(100.46))
Generation 6006: Fitness 1800.03064550624: (((t)*(4.8879))*((-t)))+(((t)*(49.8))+(100.46))
Generation 6013: Fitness 1800.030601561686: (((t)*(4.8877))*((-t)))+(((t)*(49.8))+(100.46))
Generation 6014: Fitness 1800.030477264532: (((t)*(4.8875))*((-t)))+(((t)*(49.8))+(100.44))
Generation 6607: Fitness 1800.03031917925: (((t)*(4.8974))*((-t)))+(((t)*(49.9))+(100.44))
Generation 6614: Fitness 1800.030477264532: (((t)*(4.8875))*((-t)))+(((t)*(49.8))+(100.44))
Generation 6621: Fitness 1800.029872428587: (((t)*(4.8975))*((-t)))+(((t)*(49.9))+(100.44))
Generation 6626: Fitness 1800.023505546087: (((t)*(4.8975))*((-t)))+(((t)*(49.9))+(100.41))
Generation 6636: Fitness 1800.01564948692: (((t)*(4.8975))*((-t)))+(((t)*(49.9))+(100.34))
```

```
Generation 6642: Fitness 1800.014389315708: (((t)*(4.8972))*((-t)))+(((t)*(49.9))+(100.34))
Generation 6646: Fitness 1800.014291812211: (((t)*(4.8965))*((-t)))+(((t)*(49.9))+(100.34))
Generation 6655: Fitness 1800.013913542354: (((t)*(4.8967))*((-t)))+(((t)*(49.9))+(100.34))
Generation 6656: Fitness 1800.013782611542: (((t)*(4.8972))*((-t)))+(((t)*(49.9))+(100.33))
Generation 6658: Fitness 1800.011080829711: (((t)*(4.8965))*((-t)))+(((t)*(49.9))+(100.31))
Generation 6660: Fitness 1800.010588537211: (((t)*(4.8965))*((-t)))+(((t)*(49.9))+(100.24))
Generation 6661: Fitness 1800.011080829711: (((t)*(4.8965))*((-t)))+(((t)*(49.9))+(100.31))
Generation 6668: Fitness 1800.010588537211: (((t)*(4.8965))*((-t)))+(((t)*(49.9))+(100.24))
Generation 6673: Fitness 1800.007873089163: (((t)*(4.8955))*((-t)))+(((t)*(49.9))+(100.24))
Generation 6683: Fitness 1800.00778882854: (((t)*(4.8955838))*((-t)))+(((t)*(49.9))+(100.24))
Generation 6691: Fitness 1800.007786232875: (((t)*(4.8955878))*((-t)))+(((t)*(49.9))+(100.24))
Generation 6693: Fitness 1800.007779118768: (((t)*(4.8956))*((-t)))+(((t)*(49.9))+(100.24))
Generation 6695: Fitness 1800.007636367496: (((t)*(4.8955))*((-t)))+(((t)*(49.9))+(100.22))
Generation 6708: Fitness 1800.007621956824: (((t)*(4.8954))*((-t)))+(((t)*(49.9))+(100.22))
Generation 7332: Fitness 1800.007621956824: (((t)*(49.9))+(((t)*(4.8954))*((-t))))+(100.22)
Generation 7333: Fitness 1800.007621956824: (((t)*(4.8954))*((-t)))+(((t)*(49.9))+(100.22))
Generation 7671: Fitness 1800.007621956824: (((4.8954)*(t))*((-t)))+(((t)*(49.9))+(100.22))
Generation 7733: Fitness 1800.007621956824: (((t)*(4.8954))*((-t)))+(((t)*(49.9))+(100.22))
Generation 8713: Fitness 1800.007621956824: (((t)*(t))*((-4.8954)))+(((t)*(49.9))+(100.22))
Generation 15865: Fitness 1703.59476300858: (((t)*(t))*(Tan(9457.965)))+(((t)*(49.9))+(100.22))
Generation 15866: Fitness 1800.007621956824: (((t)*(t))*((-4.8954)))+(((t)*(49.9))+(100.22))
Generation 17289: Fitness 1700.438782060947: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.9))+(100.22))
Generation 17310: Fitness 1700.052959778476: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.22))
Generation 17319: Fitness 1700.044713577615: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.27))
Generation 17326: Fitness 1700.043664337443: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.28))
Generation 17328: Fitness 1700.04281509727: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.29))
Generation 17330: Fitness 1700.041467376754: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.32))
Generation 17331: Fitness 1700.04281509727: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.29))
Generation 17332: Fitness 1700.041467376754: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.32))
Generation 17349: Fitness 1700.041418136582: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.33))
Generation 17351: Fitness 1700.041467376754: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.32))
Generation 17352: Fitness 1700.041418136582: (((t)*(t))*(Tan(7318.542)))+(((t)*(49.8))+(100.33))
Generation 18195: Fitness 1700.039347626806: (((t)*(t))*(Tan(985.09113)))+(((t)*(49.8))+(100.33))
Generation 18208: Fitness 1700.032625981862: (((t)*(t))*(Tan(985.09113)))+(((t)*(49.8))+(100.38))
Generation 18212: Fitness 1700.030904336917: (((t)*(t))*(Tan(985.09113)))+(((t)*(49.8))+(100.43))
Generation 18227: Fitness 1700.030559155736: (((t)*(t))*(Tan(985.09112)))+(((t)*(49.8))+(100.43))
Generation 18260: Fitness 1700.03048463894: (((t)*(t))*(Tan(985.09111)))+(((t)*(49.8))+(100.45))
Generation 18261: Fitness 1700.030559155736: (((t)*(t))*(Tan(985.09112)))+(((t)*(49.8))+(100.43))
Generation 18313: Fitness 1700.03048463894: (((t)*(t))*(Tan(985.09111)))+(((t)*(49.8))+(100.45))
```

13.4.3 Round 1, Simulation 3

```
Generation 1: Fitness 1.413256832614404: (Tan(Cos(Sqrt(Max(Tan(631586.49),Tanh(698387.43))))))^(Cos(phi))
Generation 4: Fitness 1.324537734067505: (ArcSin(Cos(Sqrt(Max(Tan(631586.49),Tanh(698387.43))))))^(Cos(phi))
Generation 8: Fitness 1.245658207361425: ((-ArcTan((0.6493)^(Sinh(ArcTan(Cos(phi)))))))/(ArcTan(Cos(Abs((-7308109)))))
Generation 9: Fitness 0.4458297672070926: (ArcSin(Cos(Sqrt(Max(Tan(631580.49),Tanh(258.83845))))))^(Cos(phi))
Generation 15: Fitness 0.343494000639585: (ArcSin(Cos(Sqrt(ArcTan(4415609.2)))))^(Cos(phi))
Generation 19: Fitness 0.1325987754017258: (Tan(Cos(Sqrt(Max(Tan(431586.79),Tanh(690387.43))))))^(Cos(phi))
Generation 24: Fitness 0.1005725266301924: Tan((-(-ArcSinh((ArcSin(Cos((Tanh(59.98))^(Sin(ArcTan(17.3526))))))^(Cos(phi))))))
Generation 27: Fitness 0.099077003748288: (ArcSinh(Cos(Sqrt(Min(Sinh(1.345),((1.7741342)/(Tanh((465.4)*(920.67224))))*(phi))))))^(Cos(phi))
Generation 34: Fitness 0.09878094667603396: (ArcSinh(Cos(Sqrt(Min(Sinh(1.341),((1.7741342)/(Tanh((465.4)*(920.67224))))*(phi))))))^(Cos(phi))
Generation 37: Fitness 0.09878094667603396: (ArcSinh(Cos(Sqrt(Min(Sinh(1.341),((1.7741342)/(Tanh((463.4)*(920.67224))))*(phi))))))^(Cos(phi))
Generation 38: Fitness 0.09878094667603396: (ArcSinh(Cos(Sqrt(Min(Sinh(1.341),((1.7741342)/(Tanh((463.4)*(220.67224))))*(phi))))))^(Cos(phi))
Generation 39: Fitness 0.07786134972460164: (ArcTan(Cos(Sqrt(Sqrt(Max(Tan(631586.49),ArcCos(Cos((-phi))))))))^(Cos(phi))
Generation 40: Fitness 0.05804386372467668: Tan((-(-ArcSinh((Abs(Cos(Sqrt(Max(Tan(716.82299),Tanh(8.5999607))))))^(Cos(phi))))))
Generation 44: Fitness 0.05804385767091712: Tan((-(-ArcSinh((Abs(Cos(Sqrt(Max(Tan(716.82299),Tanh(8.5699607))))))^(Cos(phi))))))
Generation 47: Fitness 0.01725418280606221: (ArcSin(Cos(ArcTan(Max(Tan(431580.59),(Tan(Cos(4776.5972)))^(Cos(phi)))))))^(Cos(phi))
Generation 48: Fitness 0.01451732637698245: (ArcSin(Cos(ArcTan(Max(Tan(431580.54),(Tan(Cos(4776.5972)))^(Cos(phi)))))))^(Cos(phi))
Generation 80: Fitness 0.01192160458931247: (Tan(Cos(ArcTan(Max(Tan(431580.54),(Tan(Cos(4776.5972)))^(Cos(phi)))))))^(Cos(phi))
Generation 84: Fitness 0.008371420168391153: Exp((-Min(0.57,Tan(Min(Cos(phi),Pi())))))
Generation 88: Fitness 0.00617478559307688: (ArcSin(Cos(ArcTan(Max(Tan(431580.54),(Tan(Cos(4776.5872)))^(Cos(phi)))))))^(Cos(phi))
Generation 98: Fitness 0.005028813401356842: (ArcSin(Cos(ArcTan(Max(Tan(631580.54),
Generation 107: Fitness 0.004405773386627589: (Tanh(Cos(Sqrt((Abs(Cos(Tanh(662587.63))))^(Cos(phi))))))^(Cos(phi))
Generation 108: Fitness 0.004405773386627589: (Tanh(Cos(Sqrt((Abs(Cos(Tanh(662587.93))))^(Cos(phi))))))^(Cos(phi))
Generation 115: Fitness 0.004405773386627589: (Tanh(Cos(Sqrt((Abs(Cos(Tanh(664587.93))))^(Cos(phi))))))^(Cos(phi))
Generation 122: Fitness 0.002188177250035437: (ArcTan(Tanh(Cos(Sqrt((Abs(Cos(Tanh(664587.93))))^(Cos(phi)))))))^(Cos(phi))
Generation 127: Fitness 0.002188177250035437: (ArcTan(Tanh(Cos(Sqrt((Abs(Cos(Tanh(666587.93))))^(Cos(phi)))))))^(Cos(phi))
Generation 130: Fitness 0.002182214712415624: (Tanh(Tanh(Cos(Sqrt((Abs(Cos(Tanh(664587.93))))^(Cos(phi)))))))^(Cos(phi))
Generation 137: Fitness 0.002182214712415624: (Tanh(Tanh(Cos(Sqrt((Abs(Cos(Tanh(664287.97))))^(Cos(phi)))))))^(Cos(phi))
Generation 141: Fitness 0.002182214712415624: (Tanh(Tanh(Cos(Sqrt((Abs(Cos(Tanh(664787.93))))^(Cos(phi)))))))^(Cos(phi))
Generation 148: Fitness 0.002182214712415624: (Tanh(Tanh(Cos(Sqrt((Abs(Cos(Tanh(764787.93))))^(Cos(phi)))))))^(Cos(phi))
Generation 151: Fitness 0.002182214712415624: (Tanh(Tanh(Cos(Sqrt((Abs(Cos(Tanh(664787.93))))^(Cos(phi)))))))^(Cos(phi))
Generation 155: Fitness 0.002182214712415624: (Tanh(Tanh(Cos(Sqrt((Abs(Cos(Tanh(614587.53))))^(Cos(phi)))))))^(Cos(phi))
Generation 156: Fitness 0.00211287556489456: (ArcTan(Tanh(Cos(Sqrt((Cos(Tanh(4.280159)))^(Cos(phi)))))))^(Cos(phi))
Generation 158: Fitness 0.002014413168036406: (ArcTan(Tanh(Cos(Sqrt((Abs(0.5393))^(Cos(phi)))))))^(Cos(phi))
Generation 166: Fitness 0.001985001877876797: (ArcTan(Tanh(Cos(Sqrt((Abs(0.5397))^(Cos(phi)))))))^(Cos(phi))
Generation 171: Fitness 0.001919970054750045: (Tanh(ArcTan(Cos(Sqrt((0.5393)^(Cos(phi)))))))^(Cos(phi))
Generation 185: Fitness 0.001885376099398342: (Tanh(ArcTan(Cos(Sqrt((0.5399)^(Cos(phi)))))))^(Cos(phi))
Generation 197: Fitness 0.00187580940094541: (Tanh(Tanh(Cos(Sqrt((0.5399)^(Cos(phi)))))))^(Cos(phi))
```

13.4.4 Round 2, Simulation 1

```
Generation 1, Farm 37, MDD 30, Fitness 341.3161909315097: ((-ArcTan(((-Exp(t)))-(80444.743))))*(92.09)
Generation 2, Farm 11, MDD 31, Fitness 307.2747652390308: Max(ArcSinh(Cosh(Sqrt(23111.48))),t)
Generation 3, Farm 10, MDD 32, Fitness 237.2783066103256: (t)+((Abs(2.3059931))^(Log(359.407)))
Generation 7, Farm 10, MDD 36, Fitness 218.8770209649776: (t)+((Abs(2.3159931))^(Log(359.427)))
Generation 8, Farm 2, MDD 37, Fitness 218.3995452089: (Max(65.63752,141.38533))-((-Abs(t)))
Generation 10, Farm 42, MDD 39, Fitness 139.450668585915: ((-(-122.50388)))+((t)*(5.567083))
Generation 12, Farm 42, MDD 41, Fitness 124.5253907014337: ((-(-122.50388)))+((t)*(ArcCosh(74.779644)))
Generation 15, Farm 42, MDD 44, Fitness 119.1730776765665: ((-(-122.50388)))+((t)*(4.1408952))
Generation 16, Farm 42, MDD 45, Fitness 113.656488356864: (3.696)*((Tan(ArcTan(t)))+(35.552))
Generation 19, Farm 42, MDD 48, Fitness 102.360802724864: (3.696)*((Tan(ArcTan(t)))+(35.052))
Generation 24, Farm 42, MDD 47, Fitness 93.65648835686402: (3.696)*((t)+(35.552))
Generation 25, Farm 42, MDD 46, Fitness 82.36080272486404: (3.696)*((t)+(35.052))
Generation 29, Farm 42, MDD 42, Fitness 79.1734545116264: (122.50381)+((t)*(4.1408952))
Generation 33, Farm 42, MDD 38, Fitness 70.4937601116264: (122.50381)+((t)*(4.5408952))
Generation 38, Farm 42, MDD 33, Fitness 70.4916886703664: (122.50581)+((t)*(4.5408052))
Generation 41, Farm 42, MDD 30, Fitness 70.49168713326061: (122.50581)+((t)*(4.5408054))
Generation 42, Farm 42, MDD 29, Fitness 70.29262259036641: (122.50581)+((t)*(4.5708052))
Generation 43, Farm 42, MDD 28, Fitness 70.16873171162641: (122.80381)+((t)*(4.5408952))
Generation 44, Farm 42, MDD 27, Fitness 70.0738365116264: (122.50381)+((t)*(4.6408952))
Generation 49, Farm 42, MDD 22, Fitness 70.0738135095464: (122.50387)+((t)*(4.6408952))
Generation 52, Farm 42, MDD 21, Fitness 70.07010223162639: (122.51381)+((t)*(4.6408952))
Generation 55, Farm 42, MDD 24, Fitness 70.0571651116264: (122.55381)+((t)*(4.6408952))
Generation 60, Farm 42, MDD 29, Fitness 70.04741578954641: (122.59387)+((t)*(4.6408952))
Generation 61, Farm 42, MDD 30, Fitness 70.0212699116264: (122.80381)+((t)*(4.5908952))
Generation 64, Farm 42, MDD 33, Fitness 70.0207070556264: (122.80581)+((t)*(4.5908952))
Generation 67, Farm 42, MDD 36, Fitness 70.0092356556264: (122.85581)+((t)*(4.5908952))
Generation 69, Farm 42, MDD 38, Fitness 70.0085005052606: (122.90381)+((t)*(4.5808054))
Generation 70, Farm 42, MDD 39, Fitness 70.0039301076264: (122.88581)+((t)*(4.5938952))
Generation 75, Farm 42, MDD 44, Fitness 70.0038613916264: (122.89381)+((t)*(4.5908952))
Generation 76, Farm 42, MDD 45, Fitness 70.00011381562641: (123.00581)+((t)*(4.5808952))
Generation 82, Farm 42, MDD 49, Fitness 70.00010786492641: (123.00581)+((t)*(4.5808452))
Generation 83, Farm 42, MDD 48, Fitness 70.0000539572264: (123.00581)+((t)*(4.5802952))
Generation 84, Farm 42, MDD 47, Fitness 70.0000107539534: (122.99387)+((t)*(4.5808982))
Generation 89, Farm 42, MDD 42, Fitness 70.0000107180776: (122.99389)+((t)*(4.5808344))
Generation 90, Farm 24, MDD 41, Fitness 70.000001: ((4.58)*(t))+(123.001)
Generation 122, Farm 42, MDD 31, Fitness 70.0000005612264: (123.00021)+((t)*(4.5800952))
Generation 125, Farm 42, MDD 34, Fitness 70.0000003627176: (123.00011)+((t)*(4.5800856))
Generation 129, Farm 42, MDD 38, Fitness 70.0000000017576: (123.00001)+((t)*(4.5800056))
Generation 135, Farm 42, MDD 44, Fitness 70.00000000152035: (123.00001)+((t)*(4.5800051))
Generation 137, Farm 42, MDD 46, Fitness 70.00000000017261: (123.00001)+((t)*(4.5800006))
Generation 138, Farm 42, MDD 47, Fitness 70.00000000011035: (123.00001)+((t)*(4.5800001))
Generation 460, Farm 47, MDD 31, Fitness 70.00000000000036: ((4.5800001)*(t))+(123)
Generation 536, Farm 8, MDD 25, Fitness 70.00000000000036: ((t)*(4.5799999))+(123)
Generation 1345, Farm 40, MDD 46, Fitness 70: ((4.58)*(t))+(123)
Generation 1347, Farm 40, MDD 44, Fitness 70: (123)+((t)*(4.58))
```

13.4.5 Round 2, Simulation 2

```
Generation 1, Farm 79, MDD 30, Fitness 11464631.24637037: Min(Pi(),((-(2829)/((-9.4377654))))*(t))
Generation 5, Farm 34, MDD 34, Fitness 9976092.502442697: (Abs((t)*((-91980.1))))/((-9.1813929E+02))
Generation 6, Farm 54, MDD 35, Fitness 8484545.104285132: (Min(t,ArcSinh((-6820415))))*(256.24151)
Generation 7, Farm 82, MDD 36, Fitness 6103918.328228698: (627.77319)-(Abs((t)*(147.0774)))
Generation 10, Farm 54, MDD 39, Fitness 4187034.180031526: (184.413)*((-Abs(Min(240648.8,t))))
Generation 12, Farm 54, MDD 41, Fitness 4187034.180031526: (184.413)*((-Abs(Min(240748.8,t))))
Generation 13, Farm 54, MDD 42, Fitness 4187034.180031526: (184.413)*((-Abs(Min(240648.8,t))))
Generation 14, Farm 76, MDD 43, Fitness 3890825.244321153: (293.2241)+((Abs(t))*((-193.76934)))
Generation 17, Farm 82, MDD 46, Fitness 3615319.997251471: (625.77319)-(Abs((t)*(197.0374)))
Generation 19, Farm 76, MDD 48, Fitness 3468048.74129628: (3410)+((Abs(t))*((-273.3823)))
Generation 21, Farm 70, MDD 50, Fitness 2339470.583660168: (-((t)*(t))*(ArcSinh(42.215436)))
Generation 23, Farm 70, MDD 48, Fitness 2144195.618787206: (-((t)*(t))*(ArcSinh(Abs(Exp(4.2512274)))))
Generation 28, Farm 70, MDD 43, Fitness 2141557.029360425: (-((t)*(t))*(ArcSinh(Abs(Exp(4.2412274)))))
Generation 29, Farm 70, MDD 42, Fitness 2135804.089793875: (-((t)*(t))*(ArcSinh(Abs(Exp(4.2146274)))))
Generation 31, Farm 70, MDD 40, Fitness 2129915.029767766: (-((t)*(t))*(ArcSinh(Abs(Sqrt(4134.358)))))
Generation 36, Farm 70, MDD 35, Fitness 2129914.911566257: (-((t)*(t))*(ArcSinh(Abs(Sqrt(4134.338)))))
Generation 38, Farm 70, MDD 33, Fitness 2129720.72693221: (-((t)*(t))*(ArcSinh(Abs(62.205436))))
Generation 40, Farm 70, MDD 31, Fitness 2129713.782116069: (-((t)*(t))*(ArcSinh(62.215436)))
Generation 41, Farm 70, MDD 30, Fitness 2129510.817274752: (-((t)*(t))*(ArcSinh(Abs(Exp(4.1512573)))))
Generation 43, Farm 82, MDD 28, Fitness 1557368.972477183: (925.77219)-(Abs((t)*(Abs((t)-(220.353)))))
Generation 55, Farm 82, MDD 24, Fitness 1557362.135280602: (925.77819)-(Abs((t)*(Abs((t)-(220.353)))))
Generation 58, Farm 82, MDD 27, Fitness 1502892.034298966: (975.77219)-(Abs((t)*(Abs((t)-(220.353)))))
Generation 60, Farm 17, MDD 29, Fitness 1256353.398199182: Min((Pi())*(((-t))*(t)),(Pi())*((74.29)*(t)))
Generation 65, Farm 70, MDD 34, Fitness 686237.021555352: (-((t)*(t))*(ArcSinh(Abs((Sqrt(4134.358))-(t)))))
Generation 72, Farm 70, MDD 41, Fitness 685307.2055248852: (-((t)*(t))*(ArcSinh(Abs((64.4)-(t)))))
Generation 74, Farm 70, MDD 43, Fitness 676153.383616341: (-((t)*(t))*(ArcSinh(Abs((Sqrt(4734.358))-(t)))))
Generation 76, Farm 70, MDD 45, Fitness 572164.6913170082: (-(((t)-(4.99626))*(t))*(ArcSinh(Abs(62.912436))))
Generation 80, Farm 70, MDD 49, Fitness 218975.4678217943: (((t)*((8)-(t)))-(t))*(4.91886)
Generation 85, Farm 70, MDD 46, Fitness 218968.0459126643: (((t)*((8)-(t)))-(t))*(4.91856)
Generation 87, Farm 66, MDD 44, Fitness 86250.3099557551: ((-((t)-(Cosh((-3.1672))))*(t)))*((-(-4.9472)))
Generation 90, Farm 15, MDD 41, Fitness 79075.2065108564: (t)*((((((43.1)-(t))-(t))-(t))-(t))-(t))
Generation 92, Farm 70, MDD 39, Fitness 42952.5884236775: (-(((t)-(8.94937))*(t))*(ArcSinh(Abs(62.922836))))
```

```
Generation 94, Farm 70, MDD 37, Fitness 12064.53762129222: (-(((t)-(9.7503652))*(t))*(ArcSinh(Abs(62.312836))))
Generation 97, Farm 70, MDD 34, Fitness 7309.149677960335: (-(((t)-(9.99626))*(t))*(ArcSinh(Abs(62.312836))))
Generation 103, Farm 70, MDD 28, Fitness 7308.144084218411: (-(((t)-(9.99626))*(t))*(ArcSinh(Abs(62.313836))))
Generation 117, Farm 70, MDD 26, Fitness 7269.494550827555: (-(((t)-(9.99626))*(t))*(ArcSinh(Abs(62.352836))))
Generation 119, Farm 70, MDD 28, Fitness 6662.206127417653: (-(((t)-(9.99626))*(t))*(ArcSinh(Abs(63.312836))))
Generation 127, Farm 15, MDD 36, Fitness 6510.934617968258: (t)*(((50.1)+((-t)))-(((-t))/(ArcTanh(Sin(50.0052)))))
Generation 130, Farm 19, MDD 39, Fitness 4723.593949945562: ((-(Tan(14.3407))*(t)))*((-(Tan(14.2347))-(t)))
Generation 137, Farm 66, MDD 46, Fitness 4573.903612235106: ((-((t)-(Cosh((-3.0242))))*(t)))*((-(-4.8472)))
Generation 139, Farm 59, MDD 48, Fitness 4537.821585291095: ((4.8442953)*(((-t))*(t)))-(((-t))*(50.17103))
Generation 142, Farm 15, MDD 49, Fitness 4506.185398891577: (t)*(((50.1)+((-t)))-(((-t))/(Sinh(Sin(50.0052)))))
Generation 143, Farm 15, MDD 48, Fitness 4497.708382891577: (t)*(((50.0052)+((-t)))-(((-t))/(Sinh(Sin(50.0052)))))
Generation 147, Farm 59, MDD 44, Fitness 4485.753310615903: ((4.8375937)*(((-t))*(t)))-(((-t))*(50.17103))
Generation 148, Farm 15, MDD 43, Fitness 4480.653240776343: (t)*(((50.0052)+((-t)))-(((-t))/(Sinh(Sin(50.0051)))))
Generation 149, Farm 15, MDD 42, Fitness 4471.180099899904: (t)*(((50.0052)+((-t)))-(((-t))/(ArcSin(Sin(50.0051)))))
Generation 156, Farm 47, MDD 35, Fitness 4467.097470415219: ((49)*(t))-(Abs((t)-(Abs(Abs((ArcSinh(63))*((t)*(t)))))))
Generation 157, Farm 19, MDD 34, Fitness 4466.94100731604: ((-(Tan(14.3409))*(t)))*((-(Tan(14.2337))-(t)))
Generation 160, Farm 47, MDD 31, Fitness 4465.097470415219: (t)-((Abs((ArcSinh(63))*((t)*(t))))-((49)*(t)))
Generation 164, Farm 10, MDD 27, Fitness 4465.005030856251: ((t)*(50.058))-(((t)*(4.84))*(t))
Generation 165, Farm 7, MDD 26, Fitness 4463.659357662756: ((t)*(4.8396362))*(((-t))-(Tan(4.8090003)))
Generation 169, Farm 7, MDD 22, Fitness 4463.656080734188: ((t)*(4.8396352))*(((-t))-(Tan(4.8090003)))
Generation 171, Farm 7, MDD 20, Fitness 4461.771869522137: ((t)*(4.8386362))*(((-t))-(Tan(4.8090003)))
Generation 180, Farm 7, MDD 29, Fitness 4461.74804726925: ((t)*(4.8383362))*(((-t))-(Tan(4.8090003)))
Generation 183, Farm 66, MDD 32, Fitness 4461.63543024469: ((-((t)-(Cosh(3.0262)))*(t)))*((-(-4.8372)))
Generation 185, Farm 7, MDD 34, Fitness 4457.244559609122: ((t)*(4.8376362))*(((-t))-(Tan(4.8088003)))
Generation 187, Farm 47, MDD 36, Fitness 3786.752181146036: (25)-((Abs((ArcSinh(63))*((t)*(t))))-((49)*(t)))
Generation 189, Farm 47, MDD 38, Fitness 2543.708830424566: ((49)*(t))-(Abs((63)-(Abs((ArcSinh(66))*((t)*(t))))))
Generation 191, Farm 47, MDD 40, Fitness 2427.626562777098: (63)-((Abs((ArcSinh(64))*((t)*(t))))-((49)*(t)))
Generation 198, Farm 40, MDD 47, Fitness 1216.14721085626: (78.66)-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 207, Farm 40, MDD 44, Fitness 1205.128960856254: (78.96)-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 208, Farm 40, MDD 43, Fitness 875.5972108562353: (98.66)-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 211, Farm 40, MDD 40, Fitness 874.5682108562386: (98.26)-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 219, Farm 40, MDD 32, Fitness 873.6109142162388: (96.7019)-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 230, Farm 40, MDD 21, Fitness 873.3957108562394: (97.26)-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 236, Farm 16, MDD 25, Fitness 592.8181170968443: (80.0052)+((t)*(((50.0012)+((-t)))-(((-t))/(Sinh(Sin(50.0072))))))
Generation 246, Farm 40, MDD 35, Fitness 77.85096085625545: ((t)+(90))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 259, Farm 40, MDD 48, Fitness 27.765960856249: ((t)+(96))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 260, Farm 40, MDD 49, Fitness 26.61091421624827: ((t)+(96.7019))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 261, Farm 40, MDD 50, Fitness 26.46217421624893: ((t)+(96.9019))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 262, Farm 40, MDD 49, Fitness 26.39571085624807: ((t)+(97.26))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 270, Farm 40, MDD 41, Fitness 26.38846085624747: ((t)+(97.16))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 294, Farm 40, MDD 23, Fitness 26.38831085624834: ((t)+(97.18))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 297, Farm 40, MDD 26, Fitness 26.38828585624787: ((t)+(97.17))-(((-t))*(((-(t)+(10)))*(4.9)))
Generation 366, Farm 40, MDD 35, Fitness 24.38828585624787: ((t)+(97.17))+((t)*(((-(t)+(10)))*(4.9)))
Generation 418, Farm 73, MDD 27, Fitness 23.92555232219899: (Abs(96.102163))+(((((-t))+(10.205))*(ArcSinh(67.201051)))*(t))
Generation 422, Farm 73, MDD 31, Fitness 23.87694801421269: (Abs(96.102163))+(((((-t))+(10.205))*(ArcSinh(67.191051)))*(t))
Generation 423, Farm 73, MDD 32, Fitness 23.78279456461182: (Abs(98.102163))+(((((-t))+(10.215))*(ArcSinh(67.201051)))*(t))
Generation 424, Farm 73, MDD 33, Fitness 19.06181698159087: (Abs(98.102003))+(((((-t))+(10.195))*(ArcSinh(67.301051)))*(t))
Generation 426, Farm 73, MDD 35, Fitness 18.93422852147602: (Abs(98.162003))+(((((-t))+(10.195))*(ArcSinh(67.301051)))*(t))
Generation 437, Farm 73, MDD 46, Fitness 18.93236950457447: (Abs(98.162903))+(((((-t))+(10.195))*(ArcSinh(67.301051)))*(t))
Generation 441, Farm 73, MDD 50, Fitness 18.85316954806642: (Abs(98.202003))+(((((-t))+(10.195))*(ArcSinh(67.301051)))*(t))
Generation 442, Farm 73, MDD 49, Fitness 18.10993238044462: (Abs(98.702003))+(((((-t))+(10.195))*(ArcSinh(67.301051)))*(t))
Generation 444, Farm 73, MDD 47, Fitness 18.0189345239142: (Abs(98.805003))+(((((-t))+(10.195))*(ArcSinh(67.301051)))*(t))
Generation 446, Farm 73, MDD 45, Fitness 17.87534264634697: (Abs(99.102003))+(((((-t))+(10.195))*(ArcSinh(67.301051)))*(t))
Generation 453, Farm 34, MDD 38, Fitness 17.60247244184263: (((t)*((t)-(ArcSinh(13482))))/((-2.0401092E-01)))+(98.5)
Generation 460, Farm 34, MDD 31, Fitness 17.5480524421114: (((t)*((t)-(ArcSinh(13452))))/((-2.0401092E-01)))+(98.5)
Generation 473, Farm 34, MDD 22, Fitness 17.53979977050757: (((t)*((t)-(ArcSinh(13452))))/((-2.0401072E-01)))+(98.5)
Generation 475, Farm 73, MDD 24, Fitness 17.52700456323299: (Abs(99.102003))+(((((-t))+(10.199))*(ArcSinh(67.301051)))*(t))
Generation 476, Farm 34, MDD 25, Fitness 17.519448619893: (((t)*((t)-(ArcSinh(13452))))/((-2.0401022E-01)))+(98.5)
Generation 484, Farm 73, MDD 33, Fitness 17.51459271518031: (Abs(99.102003))+(((((-t))+(10.199))*(ArcSinh(67.301951)))*(t))
Generation 486, Farm 73, MDD 35, Fitness 17.44979042340222: (Abs(99.102003))+(((((-t))+(10.198))*(ArcSinh(67.309051)))*(t))
Generation 491, Farm 73, MDD 40, Fitness 17.36877582892357: (Abs(99.102003))+(((((-t))+(10.198))*(ArcSinh(67.331051)))*(t))
Generation 495, Farm 34, MDD 44, Fitness 17.194174093003: (((t)*((t)-(ArcSinh(13452))))/((-2.0400012E-01)))+(98.5)
Generation 497, Farm 73, MDD 46, Fitness 16.51835712970882: (99.202003)+(((((-t))+(10.199))*(ArcSinh(67.301051)))*(t))
Generation 498, Farm 73, MDD 47, Fitness 16.47186484791183: (99.102003)+(((((-t))+(10.198))*(ArcSinh(67.307051)))*(t))
Generation 499, Farm 73, MDD 48, Fitness 16.36144811577184: (99.102003)+(((((-t))+(10.198))*(ArcSinh(67.327051)))*(t))
Generation 505, Farm 73, MDD 46, Fitness 16.18423788337111: (99.502003)+(((((-t))+(10.198))*(ArcSinh(67.327051)))*(t))
Generation 519, Farm 73, MDD 32, Fitness 16.11995000457772: (99.502003)+(((((-t))+(10.198))*(ArcSinh(67.337051)))*(t))
Generation 526, Farm 73, MDD 25, Fitness 16.11690831825959: (99.502003)+(((((-t))+(10.198))*(ArcSinh(67.347051)))*(t))
Generation 528, Farm 73, MDD 23, Fitness 16.11674404049984: (99.502303)+(((((-t))+(10.198))*(ArcSinh(67.347051)))*(t))
Generation 529, Farm 73, MDD 22, Fitness 16.10040222492064: (99.602003)+(((((-t))+(10.198))*(ArcSinh(67.337051)))*(t))
Generation 532, Farm 73, MDD 21, Fitness 16.05503159230067: (99.702003)+(((((-t))+(10.198))*(ArcSinh(67.357051)))*(t))
Generation 539, Farm 73, MDD 28, Fitness 16.03751810232671: (99.752003)+(((((-t))+(10.198))*(ArcSinh(67.357051)))*(t))
Generation 552, Farm 73, MDD 41, Fitness 16.03633302312875: (99.756003)+(((((-t))+(10.198))*(ArcSinh(67.357051)))*(t))
Generation 556, Farm 73, MDD 45, Fitness 16.01497763240465: (99.902003)+(((((-t))+(10.198))*(ArcSinh(67.357051)))*(t))
Generation 557, Farm 73, MDD 46, Fitness 16.01496913133498: (99.902003)+(((((-t))+(10.198))*(ArcSinh(67.357058)))*(t))
Generation 560, Farm 70, MDD 49, Fitness 15.50967451667502: (99.122911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.302116)))
Generation 564, Farm 70, MDD 47, Fitness 15.50967280120823: (99.122921)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.302116)))
Generation 569, Farm 70, MDD 42, Fitness 15.50128552421977: (99.122921)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.302716)))
Generation 575, Farm 70, MDD 36, Fitness 15.39870798367659: (99.122911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.312116)))
Generation 576, Farm 70, MDD 35, Fitness 15.36064498937293: (99.122911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
Generation 584, Farm 70, MDD 27, Fitness 15.36064034399138: (99.122916)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
Generation 587, Farm 70, MDD 24, Fitness 15.34903381019148: (99.122921)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.322116)))
Generation 590, Farm 70, MDD 21, Fitness 15.30850011058923: (99.182911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
Generation 596, Farm 70, MDD 25, Fitness 15.21482872676044: (99.322911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
```

```
Generation 597, Farm 70, MDD 26, Fitness 15.20028807022932: (99.822911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
Generation 602, Farm 70, MDD 31, Fitness 15.14901246414809: (99.522911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
Generation 604, Farm 70, MDD 33, Fitness 15.14610468746039: (99.622916)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
Generation 608, Farm 70, MDD 37, Fitness 15.14486758536433: (99.582911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.332116)))
Generation 610, Farm 70, MDD 39, Fitness 15.13062473364518: (99.822911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.338116)))
Generation 611, Farm 70, MDD 40, Fitness 15.09643375259013: (99.822911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.342116)))
Generation 615, Farm 70, MDD 44, Fitness 15.09273666133593: (99.622911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.342116)))
Generation 618, Farm 70, MDD 47, Fitness 15.05382292235689: (99.822911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.352116)))
Generation 619, Farm 70, MDD 48, Fitness 15.05382275283038: (99.822916)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.352116)))
Generation 621, Farm 70, MDD 50, Fitness 15.0537056133495: (99.852911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.352116)))
Generation 624, Farm 70, MDD 47, Fitness 15.05368538158972: (99.852111)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.352116)))
Generation 634, Farm 70, MDD 37, Fitness 15.0526475749456: (99.822911)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.354116)))
Generation 638, Farm 70, MDD 33, Fitness 15.05083544129839: (99.852111)-((((t)-(ArcSinh(13435.8)))*(t))*(ArcSinh(67.356116)))
Generation 640, Farm 70, MDD 31, Fitness 15.04993109434364: (99.852911)-((((t)-(ArcSinh(13434.8)))*(t))*(ArcSinh(67.352126)))
Generation 642, Farm 70, MDD 29, Fitness 15.04664010093717: (99.982911)-((((t)-(ArcSinh(13425.8)))*(t))*(ArcSinh(67.352116)))
Generation 645, Farm 70, MDD 26, Fitness 15.0465408307599: (99.822916)-((((t)-(ArcSinh(13433.8)))*(t))*(ArcSinh(67.352116)))
Generation 647, Farm 70, MDD 24, Fitness 15.02179056970178: (99.822316)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.352116)))
Generation 649, Farm 70, MDD 22, Fitness 15.01126274711836: (99.852111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.356116)))
Generation 656, Farm 70, MDD 25, Fitness 15.01126198263723: (99.852121)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.356116)))
Generation 664, Farm 70, MDD 33, Fitness 15.01097291466798: (99.856111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.356116)))
Generation 665, Farm 70, MDD 34, Fitness 15.0104549625887: (99.852111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.356816)))
Generation 668, Farm 70, MDD 37, Fitness 15.00991237555543: (99.852111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359116)))
Generation 669, Farm 70, MDD 38, Fitness 15.00975046422368: (99.852111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.358116)))
Generation 674, Farm 70, MDD 43, Fitness 15.00974919502598: (99.852121)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.358116)))
Generation 678, Farm 70, MDD 47, Fitness 15.00761099960211: (99.872121)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.358116)))
Generation 681, Farm 70, MDD 50, Fitness 15.00705748710434: (99.952111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.358116)))
Generation 684, Farm 70, MDD 47, Fitness 15.00619496933467: (99.882111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359816)))
Generation 698, Farm 70, MDD 33, Fitness 15.00618822359528: (99.882111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359416)))
Generation 699, Farm 70, MDD 32, Fitness 15.00618399638435: (99.882211)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359816)))
Generation 700, Farm 70, MDD 31, Fitness 15.00611857868254: (99.882811)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359816)))
Generation 704, Farm 70, MDD 27, Fitness 15.00599608252821: (99.932111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.358116)))
Generation 705, Farm 70, MDD 26, Fitness 15.00597931032863: (99.884111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359816)))
Generation 707, Farm 70, MDD 24, Fitness 15.00534009126269: (99.952111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.358816)))
Generation 710, Farm 70, MDD 21, Fitness 15.00519633780384: (99.892111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359819)))
Generation 720, Farm 70, MDD 29, Fitness 15.0051877013543: (99.892211)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359816)))
Generation 721, Farm 70, MDD 30, Fitness 15.00518045493219: (99.952111)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.358888)))
Generation 723, Farm 70, MDD 32, Fitness 15.00321280850223: (99.942811)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359816)))
Generation 736, Farm 70, MDD 45, Fitness 15.00320942735309: (99.942511)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359816)))
Generation 738, Farm 70, MDD 47, Fitness 15.00314394901008: (99.942811)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359866)))
Generation 739, Farm 70, MDD 48, Fitness 15.00311683362208: (99.942811)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359886)))
Generation 740, Farm 70, MDD 49, Fitness 15.0031033677299: (99.942811)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359896)))
Generation 743, Farm 70, MDD 48, Fitness 15.00310327231272: (99.942801)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359896)))
Generation 744, Farm 70, MDD 47, Fitness 15.00310059221603: (99.942511)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359896)))
Generation 748, Farm 70, MDD 43, Fitness 15.00309791328509: (99.942201)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359896)))
Generation 757, Farm 70, MDD 34, Fitness 15.00307661954894: (99.942811)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359916)))
Generation 760, Farm 70, MDD 31, Fitness 15.00297207486692: (99.942811)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359996)))
Generation 763, Farm 70, MDD 28, Fitness 15.00296815974304: (99.942201)-((((t)-(ArcSinh(13415.8)))*(t))*(ArcSinh(67.359996)))
Generation 775, Farm 70, MDD 24, Fitness 15.00295857500657: (99.942201)-((((t)-(ArcSinh(13415.3)))*(t))*(ArcSinh(67.359996)))
Generation 778, Farm 70, MDD 27, Fitness 15.0029559501039: (99.942201)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359996)))
Generation 784, Farm 70, MDD 33, Fitness 15.002955596177: (99.942201)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359996)))
Generation 792, Farm 70, MDD 41, Fitness 15.00295203919289: (99.942201)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359999)))
Generation 799, Farm 70, MDD 48, Fitness 15.00295114186807: (99.941201)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359996)))
Generation 802, Farm 70, MDD 49, Fitness 15.00295078794116: (99.941201)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359996)))
Generation 804, Farm 70, MDD 47, Fitness 15.00294752539642: (99.939201)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359996)))
Generation 810, Farm 70, MDD 41, Fitness 15.0029473066613: (99.941201)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359999)))
Generation 825, Farm 70, MDD 26, Fitness 15.00294717081692: (99.940201)-((((t)-(ArcSinh(13415.3)))*(t))*(ArcSinh(67.359999)))
Generation 826, Farm 70, MDD 25, Fitness 15.00294698094973: (99.941201)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 830, Farm 70, MDD 21, Fitness 15.00294506891893: (99.939201)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359998)))
Generation 834, Farm 70, MDD 23, Fitness 15.00294384159815: (99.939201)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359999)))
Generation 849, Farm 70, MDD 38, Fitness 15.00294383664882: (99.939221)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359999)))
Generation 851, Farm 70, MDD 40, Fitness 15.00294382777537: (99.939271)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359999)))
Generation 854, Farm 70, MDD 43, Fitness 15.00294382562603: (99.939291)-((((t)-(ArcSinh(13415.4)))*(t))*(ArcSinh(67.359999)))
Generation 859, Farm 70, MDD 48, Fitness 15.00294351588658: (99.939201)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 864, Farm 70, MDD 47, Fitness 15.00294350206379: (99.939271)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 875, Farm 70, MDD 36, Fitness 15.00294350176214: (99.939271)-((((t)-(ArcCosh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 877, Farm 70, MDD 34, Fitness 15.00294349991445: (99.939291)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 878, Farm 70, MDD 33, Fitness 15.00294349913977: (99.939301)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 888, Farm 70, MDD 23, Fitness 15.00294349882742: (99.939306)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 893, Farm 70, MDD 22, Fitness 15.00294349877096: (99.939307)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 896, Farm 70, MDD 25, Fitness 15.00294349819038: (99.939321)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 906, Farm 70, MDD 35, Fitness 15.00294349812865: (99.939346)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 908, Farm 70, MDD 37, Fitness 15.00294349796471: (99.939351)-((((t)-(ArcCosh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 923, Farm 70, MDD 48, Fitness 15.00294349788873: (99.939321)-((((t)-(ArcCosh(13415.5)))*(t))*(ArcSinh(67.359999)))
Generation 925, Farm 70, MDD 46, Fitness 15.00288617816767: (99.939301)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.363939)))
Generation 940, Farm 70, MDD 31, Fitness 15.00212921605163: (99.939301)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.363139)))
Generation 947, Farm 70, MDD 24, Fitness 15.00199128342517: (99.949301)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.363939)))
Generation 949, Farm 70, MDD 22, Fitness 15.00129638868267: (99.959301)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.363939)))
Generation 953, Farm 70, MDD 22, Fitness 15.00041170445516: (99.989301)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.363939)))
Generation 969, Farm 70, MDD 38, Fitness 15.00040893445311: (99.989301)-((((t)-(ArcSinh(13415.5)))*(t))*(ArcSinh(67.363989)))
Generation 973, Farm 70, MDD 42, Fitness 15.00010861028793: (99.989301)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.363939)))
Generation 983, Farm 70, MDD 48, Fitness 15.0000964349728: (99.989301)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.363989)))
Generation 990, Farm 70, MDD 41, Fitness 15.00009418348569: (99.989301)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.363999)))
Generation 1001, Farm 70, MDD 30, Fitness 15.00008539107281: (99.989301)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.363989)))
Generation 1007, Farm 70, MDD 24, Fitness 15.0000834217594: (99.989301)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.363999)))
```

```
Generation 1017, Farm 70, MDD 26, Fitness 15.00008333954709: (99.989391)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.363999)))
Generation 1019, Farm 70, MDD 28, Fitness 15.00008317002439: (99.989801)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.363999)))
Generation 1040, Farm 70, MDD 49, Fitness 15.00008317002193: (99.989802)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.363999)))
Generation 1051, Farm 70, MDD 40, Fitness 15.00008317002146: (99.989803)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.363999)))
Generation 1141, Farm 70, MDD 30, Fitness 15.00003026080965: (99.999803)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.364999)))
Generation 1152, Farm 70, MDD 41, Fitness 15.00002104731: (99.999803)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.364929)))
Generation 1158, Farm 70, MDD 47, Fitness 15.00001635220681: (99.999803)-((((t)-(ArcSinh(13413.6)))*(t))*(ArcSinh(67.364999)))
Generation 1159, Farm 70, MDD 48, Fitness 15.00001579470464: (99.999803)-((((t)-(ArcSinh(13413.2)))*(t))*(ArcSinh(67.364999)))
Generation 1160, Farm 70, MDD 49, Fitness 15.00001188489397: (99.999803)-((((t)-(ArcSinh(13413.8)))*(t))*(ArcSinh(67.364699)))
Generation 1168, Farm 70, MDD 43, Fitness 15.00000843474116: (99.999809)-((((t)-(ArcSinh(13413.6)))*(t))*(ArcSinh(67.364929)))
Generation 1174, Farm 70, MDD 37, Fitness 15.00000660500864: (99.999803)-((((t)-(ArcSinh(13413.3)))*(t))*(ArcSinh(67.364929)))
Generation 1180, Farm 70, MDD 31, Fitness 15.00000633411009: (99.999883)-((((t)-(ArcSinh(13413.3)))*(t))*(ArcSinh(67.364929)))
Generation 1183, Farm 70, MDD 28, Fitness 15.00000598904657: (99.999803)-((((t)-(ArcSinh(13413.6)))*(t))*(ArcSinh(67.364899)))
Generation 1187, Farm 70, MDD 24, Fitness 15.00000289452355: (99.999803)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.364699)))
Generation 1194, Farm 70, MDD 23, Fitness 15.00000217463162: (99.999809)-((((t)-(ArcSinh(13413.6)))*(t))*(ArcSinh(67.364822)))
Generation 1195, Farm 70, MDD 24, Fitness 15.00000007910999: (99.999803)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1210, Farm 70, MDD 39, Fitness 15.00000007892504: (99.999804)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1211, Farm 70, MDD 40, Fitness 15.00000007063389: (99.999883)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1218, Farm 70, MDD 47, Fitness 15.00000007047437: (99.999893)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1224, Farm 70, MDD 47, Fitness 15.00000007046942: (99.999894)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1234, Farm 70, MDD 37, Fitness 15.00000007046552: (99.999896)-((((t)-(ArcSinh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1251, Farm 70, MDD 20, Fitness 15.00000007027204: (99.999894)-((((t)-(ArcCosh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1261, Farm 70, MDD 30, Fitness 15.00000007026814: (99.999896)-((((t)-(ArcCosh(13413.5)))*(t))*(ArcSinh(67.364799)))
Generation 1275, Farm 44, MDD 44, Fitness 14.94245856377768: ((t)*(((t)/(Cos(77.174401)))+(50.000001)))+(99.093945)
Generation 1283, Farm 44, MDD 48, Fitness 14.07563277778343: ((t)*(((t)/(Cos(77.174401)))+(50.000001)))+(99.793945)
Generation 1287, Farm 44, MDD 44, Fitness 14.01942138858843: ((t)*(((t)/(Cos(77.174409)))+(50.000001)))+(99.793945)
Generation 1306, Farm 44, MDD 25, Fitness 14.00034707926058: ((t)*(((t)/(Cos(77.174405)))+(50.000001)))+(99.993945)
Generation 1312, Farm 44, MDD 21, Fitness 14.00034664500449: ((t)*(((t)/(Cos(77.174405)))+(50.000001)))+(99.993925)
Generation 1327, Farm 44, MDD 36, Fitness 14.00034644092642: ((t)*(((t)/(Cos(77.174405)))+(50.000004)))+(99.993915)
Generation 1328, Farm 44, MDD 37, Fitness 14.00034642817643: ((t)*(((t)/(Cos(77.174405)))+(50.000001)))+(99.993915)
Generation 1329, Farm 44, MDD 38, Fitness 14.00033400207172: ((t)*(((t)/(Cos(77.174405)))+(50.000004)))+(99.993325)
Generation 1330, Farm 44, MDD 39, Fitness 14.0003076136514: ((t)*(((t)/(Cos(77.174405)))+(50.000001)))+(99.991945)
Generation 1335, Farm 44, MDD 44, Fitness 14.00009115716388: ((t)*(((t)/(Cos(77.174404)))+(50.000001)))+(99.993945)
Generation 1340, Farm 44, MDD 49, Fitness 14.0000174744898: ((t)*(((t)/(Cos(77.174404)))+(50.000001)))+(99.999945)
Generation 1364, Farm 44, MDD 27, Fitness 14.00001734928088: ((t)*(((t)/(Cos(77.174404)))+(50.000001)))+(99.999965)
Generation 1374, Farm 44, MDD 23, Fitness 14.00001728697642: ((t)*(((t)/(Cos(77.174404)))+(50.000001)))+(99.999975)
Generation 1375, Farm 44, MDD 24, Fitness 14.00001722487197: ((t)*(((t)/(Cos(77.174404)))+(50.000001)))+(99.999985)
Generation 1376, Farm 44, MDD 25, Fitness 14.00001716296751: ((t)*(((t)/(Cos(77.174404)))+(50.000001)))+(99.999995)
Generation 1399, Farm 44, MDD 48, Fitness 14.00001713826173: ((t)*(((t)/(Cos(77.174404)))+(50.000001)))+(99.999999)
Generation 1693, Farm 44, MDD 42, Fitness 14.00001713741173: ((t)*(((t)/(Cos(77.174404)))+(50)))+(99.999999)
Generation 1883, Farm 13, MDD 48, Fitness 13.42441416475117: (99.173749)+(((50)-((t)*(4.90309)))*(t))
Generation 1887, Farm 13, MDD 44, Fitness 13.42435151715096: (99.173799)+(((50)-((t)*(4.90309)))*(t))
Generation 1889, Farm 13, MDD 42, Fitness 13.31508978519102: (99.173729)+(((50)-((t)*(4.90269)))*(t))
Generation 1901, Farm 13, MDD 30, Fitness 13.21381376475097: (99.373749)+(((50)-((t)*(4.90309)))*(t))
Generation 1906, Farm 13, MDD 25, Fitness 13.13851356475104: (99.473749)+(((50)-((t)*(4.90309)))*(t))
Generation 1907, Farm 13, MDD 24, Fitness 13.08321336475099: (99.573749)+(((50)-((t)*(4.90309)))*(t))
Generation 1915, Farm 13, MDD 24, Fitness 13.08141735675101: (99.577749)+(((50)-((t)*(4.90309)))*(t))
Generation 1916, Farm 13, MDD 25, Fitness 13.06823176779102: (99.773429)+(((50)-((t)*(4.90299)))*(t))
Generation 1918, Farm 13, MDD 27, Fitness 13.00907885559105: (99.973779)+(((50)-((t)*(4.90339)))*(t))
Generation 1930, Farm 13, MDD 39, Fitness 13.00844308759101: (99.977779)+(((50)-((t)*(4.90339)))*(t))
Generation 1933, Farm 13, MDD 42, Fitness 13.00707270259111: (99.973779)+(((50)-((t)*(4.90338)))*(t))
Generation 1934, Farm 13, MDD 43, Fitness 13.00623305059112: (99.979779)+(((50)-((t)*(4.90338)))*(t))
Generation 1935, Farm 13, MDD 44, Fitness 13.00049717059101: (99.973779)+(((50)-((t)*(4.90332)))*(t))
Generation 1944, Farm 13, MDD 47, Fitness 13.00031147959101: (99.973779)+(((50)-((t)*(4.90331)))*(t))
Generation 1948, Farm 13, MDD 43, Fitness 13.00020353575102: (99.990499)+(((50)-((t)*(4.90333)))*(t))
Generation 1950, Farm 13, MDD 41, Fitness 13.00017173043099: (99.993709)+(((50)-((t)*(4.90331)))*(t))
Generation 1955, Farm 13, MDD 36, Fitness 13.000142059591: (99.983779)+(((50)-((t)*(4.90331)))*(t))
Generation 1963, Farm 13, MDD 28, Fitness 13.000018611431: (99.993709)+(((50)-((t)*(4.90332)))*(t))
Generation 1971, Farm 13, MDD 20, Fitness 13.000018330591: (99.993779)+(((50)-((t)*(4.90332)))*(t))
Generation 1989, Farm 13, MDD 38, Fitness 13.000015357775: (99.996705)+(((50)-((t)*(4.90332)))*(t))
Generation 1994, Farm 13, MDD 43, Fitness 13.000014446591: (99.995779)+(((50)-((t)*(4.90332)))*(t))
Generation 2001, Farm 13, MDD 50, Fitness 13.000014445871: (99.995739)+(((50)-((t)*(4.90332)))*(t))
Generation 2012, Farm 13, MDD 39, Fitness 13.000014445751: (99.995749)+(((50)-((t)*(4.90332)))*(t))
Generation 4416, Farm 0, MDD 35, Fitness 13.0000096000825: (99.9965)-((t)*(((t)*(4.9033245))-(50)))
Generation 4426, Farm 0, MDD 25, Fitness 13.0000001500825: (99.9995)-((t)*(((t)*(4.9033245))-(50)))
Generation 4449, Farm 0, MDD 38, Fitness 13.0000001120297: (99.9995)-((t)*(((t)*(4.9033247))-(50)))
Generation 4459, Farm 0, MDD 48, Fitness 13.0000000730297: (99.9996)-((t)*(((t)*(4.9033247))-(50)))
Generation 4460, Farm 0, MDD 49, Fitness 13.0000000520033: (99.9997)-((t)*(((t)*(4.9033249))-(50)))
Generation 4468, Farm 0, MDD 43, Fitness 13.0000000060033: (99.9999)-((t)*(((t)*(4.9033249))-(50)))
```

13.4.6 Round 2, Simulation 3

```
Generation 1, Farm 41, MDD 40, Fitness 0.7675232041447567: (ArcTan(67099.5))-((Tan(((-62))/(3493670.3)))+(Cos(phi)))
Generation 2, Farm 41, MDD 41, Fitness 0.7675231494360495: (ArcTan(67099.5))-((Tan(((-63))/(3493670.3)))+(Cos(phi)))
Generation 5, Farm 41, MDD 44, Fitness 0.7675231347935417: (ArcTan(67099.5))-((Tan(((-62))/(3423670.3)))+(Cos(phi)))
Generation 6, Farm 20, MDD 45, Fitness 0.5754971752715802: ArcCos(Sin((Tan(Abs(Cosh((-ArcSinh((Sqrt(1.1883459E-39))*(561.09)))))))-(phi)))
Generation 9, Farm 13, MDD 48, Fitness 0.5567781306052719: ArcCos(Sin((ArcCos(Cos(phi)))+((-(-33.072448)))))
Generation 10, Farm 28, MDD 49, Fitness 0.3340531327706943: ((-(-3238.5507)))^((Log(Sin(Tan(35.318281))))*(Cos(phi)))
Generation 13, Farm 20, MDD 52, Fitness 0.1612652448588457: (Sqrt(Cos((-303.1053))))^(Cos(Abs(phi)))
Generation 17, Farm 1, MDD 56, Fitness 0.1314930596839567: (Abs(ArcTan(Cos(Sqrt(ArcTan((1.0571868)+(phi)))))))^(Tan(Cos(phi)))
Generation 20, Farm 48, MDD 59, Fitness 0.1214259065523106: (Tanh(611833.44))/((ArcTan(Cos(phi)))+((2069536.3)^((29777.01)^((-Sqrt(phi))))))
Generation 25, Farm 48, MDD 56, Fitness 0.1210328100325869: (Tanh(611833.44))/((ArcTan(Cos(phi)))+((46.807)^((29777.01)^((-Sqrt(phi))))))
```

```
Generation 26, Farm 1, MDD 55, Fitness 0.1210180351746581: (Abs(Sin(Cos(Sqrt(ArcTan((2.0571868)+(phi)))))))^(Tan(Cos(phi)))
Generation 29, Farm 48, MDD 52, Fitness 0.1188767188628275: (Tanh(611833.44))/((ArcTan(Cos(phi)))+((5.158)^((29777.01)^((-Sqrt(phi))))))
Generation 30, Farm 48, MDD 51, Fitness 0.1114959055153991: (Tanh(611833.44))/((ArcTan(Cos(phi)))+((2069536.3)^((19676.64)^((-Pi())))))
Generation 31, Farm 37, MDD 50, Fitness 0.08697802540440702: (Sin((-Exp(48.6515))))^(Tan(Cos(phi)))
Generation 36, Farm 48, MDD 45, Fitness 0.07649590551542736: (Tanh(611833.44))/((ArcTan(Cos(phi)))+(Cos(5.927659E-65)))
Generation 45, Farm 48, MDD 36, Fitness 0.07649590551542736: (Tanh(396.8))/((ArcTan(Cos(phi)))+(Tanh(611833.44)))
Generation 47, Farm 22, MDD 34, Fitness 0.05784888347281913: (Cosh(ArcCos(Cos(phi))))*(0.4667326)
Generation 51, Farm 22, MDD 30, Fitness 0.0576976282186721: (Cosh(ArcCos(Cos(phi))))*(0.4662326)
Generation 61, Farm 22, MDD 20, Fitness 0.05720143788504241: (Cosh(ArcCos(Cos(phi))))*(0.4642326)
Generation 63, Farm 22, MDD 22, Fitness 0.05682606900466208: (Cosh(ArcCos(Cos(phi))))*(0.4617326)
Generation 66, Farm 22, MDD 25, Fitness 0.05682604028695783: (Cosh(ArcCos(Cos(phi))))*(0.4617323)
Generation 67, Farm 22, MDD 26, Fitness 0.05682511385916902: (Cosh(ArcCos(Cos(phi))))*(0.4617226)
Generation 68, Farm 22, MDD 27, Fitness 0.05675210293058096: (Cosh(ArcCos(Cos(phi))))*(0.4607326)
Generation 71, Farm 22, MDD 30, Fitness 0.05673144449598418: (Cosh(ArcCos(Cos(phi))))*(0.4602326)
Generation 76, Farm 22, MDD 35, Fitness 0.0567262283812683: (Cosh(ArcCos(Cos(phi))))*(0.4600326)
Generation 77, Farm 22, MDD 36, Fitness 0.05672277683167819: (Cosh(ArcCos(Cos(phi))))*(0.4592326)
Generation 80, Farm 22, MDD 39, Fitness 0.05672252356868711: (Cosh(ArcCos(Cos(phi))))*(0.4592526)
Generation 88, Farm 22, MDD 47, Fitness 0.05672102002461858: (Cosh(ArcCos(Cos(phi))))*(0.4596326)
Generation 93, Farm 22, MDD 52, Fitness 0.05672094864878637: (Cosh(ArcCos(Cos(phi))))*(0.4594526)
Generation 98, Farm 41, MDD 57, Fitness 0.05244684983819263: (ArcTan(3.91463))/((Cos(phi))+(ArcTan(Pi())))
Generation 100, Farm 41, MDD 59, Fitness 0.05081648653509909: (ArcTan(3.61162))/((Cos(phi))+(ArcTan(Pi())))
Generation 104, Farm 41, MDD 57, Fitness 0.05081461779096096: (ArcTan(3.61062))/((Cos(phi))+(ArcTan(Pi())))
Generation 109, Farm 41, MDD 52, Fitness 0.05078939009252183: (ArcTan(3.4965026))/((Cos(phi))+(ArcTan(Pi())))
Generation 111, Farm 41, MDD 50, Fitness 0.05076714420853945: (ArcTan(3.51469))/((Cos(phi))+(ArcTan(Pi())))
Generation 118, Farm 41, MDD 43, Fitness 0.05075463647403389: (ArcTan(3.53469))/((Cos(phi))+(ArcTan(Pi())))
Generation 119, Farm 41, MDD 42, Fitness 0.05075454228569102: (ArcTan(3.53499))/((Cos(phi))+(ArcTan(Pi())))
Generation 120, Farm 41, MDD 41, Fitness 0.05075296178556268: (ArcTan(3.54469))/((Cos(phi))+(ArcTan(Pi())))
Generation 122, Farm 41, MDD 39, Fitness 0.05073028110546576: (ArcTan(2.61969))/((Cos(phi))+(ArcTan(2.91861)))
Generation 132, Farm 41, MDD 29, Fitness 0.05044646423395534: (ArcTan(2.69969))/((Cos(phi))+(ArcTan(2.91861)))
Generation 137, Farm 41, MDD 24, Fitness 0.05044315751973973: (ArcTan(2.69969))/((Cos(phi))+(ArcTan(2.91891)))
Generation 147, Farm 41, MDD 26, Fitness 0.05043582064219271: (ArcTan(2.69969))/((Cos(phi))+(ArcTan(2.91961)))
Generation 152, Farm 41, MDD 31, Fitness 0.05015585290284278: (ArcTan(2.81969))/((Cos(phi))+(ArcTan(2.96841)))
Generation 159, Farm 41, MDD 38, Fitness 0.05008906134310215: (ArcTan(2.91861))/((Cos(phi))+(ArcTan(2.99861)))
Generation 162, Farm 41, MDD 41, Fitness 0.05008830603647325: (ArcTan(2.86969))/((Cos(phi))+(ArcTan(2.96841)))
Generation 165, Farm 41, MDD 44, Fitness 0.05008496686894649: (ArcTan(2.91861))/((Cos(phi))+(ArcTan(2.99821)))
Generation 168, Farm 41, MDD 47, Fitness 0.05008396534701876: (ArcTan(2.91861))/((Cos(phi))+(ArcTan(2.99811)))
Generation 170, Farm 41, MDD 49, Fitness 0.05000600017583385: (ArcTan(2.96861))/((Cos(phi))+(ArcTan(2.99861)))
Generation 173, Farm 41, MDD 52, Fitness 0.05000581403586482: (ArcTan(2.96841))/((Cos(phi))+(ArcTan(2.99891)))
Generation 181, Farm 41, MDD 60, Fitness 0.05000560193640698: (ArcTan(2.96861))/((Cos(phi))+(ArcTan(2.99961)))
Generation 188, Farm 41, MDD 53, Fitness 0.05000553717208552: (ArcTan(2.96896))/((Cos(phi))+(ArcTan(2.99991)))
Generation 191, Farm 41, MDD 50, Fitness 0.05000551763188619: (ArcTan(2.96891))/((Cos(phi))+(ArcTan(2.99961)))
Generation 197, Farm 41, MDD 44, Fitness 0.05000534257826467: (ArcTan(2.96961))/((Cos(phi))+(ArcTan(2.99961)))
Generation 201, Farm 41, MDD 40, Fitness 0.05000524378928234: (ArcTan(2.96992))/((Cos(phi))+(ArcTan(2.99991)))
Generation 202, Farm 41, MDD 39, Fitness 0.05000523280184811: (ArcTan(2.96996))/((Cos(phi))+(ArcTan(2.99991)))
Generation 212, Farm 41, MDD 29, Fitness 0.05000516757823563: (ArcTan(2.97861))/((Cos(phi))+(ArcTan(2.99991)))
Generation 215, Farm 41, MDD 26, Fitness 0.05000512971528777: (ArcTan(2.97846))/((Cos(phi))+(ArcTan(2.99991)))
Generation 217, Farm 41, MDD 24, Fitness 0.05000470054762961: (ArcTan(2.97596))/((Cos(phi))+(ArcTan(2.99991)))
Generation 227, Farm 41, MDD 26, Fitness 0.05000468269782146: (ArcTan(2.97576))/((Cos(phi))+(ArcTan(2.99991)))
Generation 232, Farm 41, MDD 31, Fitness 0.05000464382972351: (ArcTan(2.97516))/((Cos(phi))+(ArcTan(2.99991)))
Generation 233, Farm 41, MDD 32, Fitness 0.05000463629399543: (ArcTan(2.97596))/((Cos(phi))+(ArcTan(2.99996)))
Generation 236, Farm 41, MDD 35, Fitness 0.05000462756212561: (ArcTan(2.97461))/((Cos(phi))+(ArcTan(2.99991)))
Generation 237, Farm 41, MDD 36, Fitness 0.05000459227740938: (ArcTan(2.97526))/((Cos(phi))+(ArcTan(2.99996)))
Generation 242, Farm 41, MDD 41, Fitness 0.05000457826334272: (ArcTan(2.97461))/((Cos(phi))+(ArcTan(2.99996)))
Generation 248, Farm 41, MDD 47, Fitness 0.05000455229149427: (ArcTan(2.97499))/((Cos(phi))+(ArcTan(2.99999)))
Generation 251, Farm 41, MDD 50, Fitness 0.05000454976692578: (ArcTan(2.97461))/((Cos(phi))+(ArcTan(2.99999)))
Generation 259, Farm 41, MDD 58, Fitness 0.05000454957343498: (ArcTan(2.97471))/((Cos(phi))+(ArcTan(2.99999)))
Generation 267, Farm 41, MDD 54, Fitness 0.0500045495631042: (ArcTan(2.97469))/((Cos(phi))+(ArcTan(2.99999)))
Generation 388, Farm 41, MDD 27, Fitness 0.04506461868488816: (1.242)/((Cos(phi))+(ArcTan(2.99999)))
Generation 393, Farm 41, MDD 32, Fitness 0.04502471432312263: (1.242)/((Cos(phi))+(ArcTan(2.99299)))
Generation 399, Farm 41, MDD 38, Fitness 0.04501624456644356: (1.243)/((Cos(phi))+(ArcTan(2.99299)))
Generation 402, Farm 41, MDD 41, Fitness 0.04501127659919948: (1.248)/((Cos(phi))+(ArcTan(2.99999)))
Generation 404, Farm 41, MDD 43, Fitness 0.04501119764293641: (1.245)/((Cos(phi))+(ArcTan(2.99999)))
Generation 408, Farm 41, MDD 47, Fitness 0.04501115171093922: (1.245)/((Cos(phi))+(ArcTan(2.99997)))
Generation 409, Farm 41, MDD 48, Fitness 0.04500527951744467: (1.246)/((Cos(phi))+(ArcTan(2.99999)))
Generation 410, Farm 41, MDD 49, Fitness 0.04500527219170252: (1.246)/((Cos(phi))+(ArcTan(2.99979)))
Generation 412, Farm 41, MDD 51, Fitness 0.04500506712632812: (ArcTan(2.97469))/((Cos(phi))+(1.249))
Generation 416, Farm 41, MDD 55, Fitness 0.04006075834115534: (1.242)/((Cos(phi))+(1.249))
Generation 420, Farm 41, MDD 59, Fitness 0.04002132028702287: (1.242)/((Cos(phi))+(1.248))
Generation 421, Farm 41, MDD 60, Fitness 0.04002038130323288: (1.243)/((Cos(phi))+(1.248))
Generation 424, Farm 41, MDD 57, Fitness 0.04001326649639184: (1.248)/((Cos(phi))+(1.249))
Generation 426, Farm 41, MDD 55, Fitness 0.04000531306414709: (1.246)/((Cos(phi))+(1.249))
Generation 640, Farm 45, MDD 39, Fitness 0.04000527941741799: (1.254)/((Cos(phi))+(1.251))
Generation 1638, Farm 41, MDD 43, Fitness 0.04000527520386239: (1.246)/((Cos(phi))+(1.2490206))
Generation 1640, Farm 41, MDD 41, Fitness 0.04000527066158163: (1.246)/((Cos(phi))+(1.2490296))
Generation 1653, Farm 41, MDD 28, Fitness 0.04000527063580721: (1.246)/((Cos(phi))+(1.2490299))
Generation 1655, Farm 41, MDD 26, Fitness 0.04000284733563082: (1.249)/((Cos(phi))+(1.2495206))
Generation 1659, Farm 41, MDD 22, Fitness 0.04000284308266235: (1.249)/((Cos(phi))+(1.2495208))
Generation 1668, Farm 41, MDD 27, Fitness 0.04000151728438074: (1.249)/((Cos(phi))+(1.2499206))
Generation 1673, Farm 41, MDD 32, Fitness 0.04000006870585508: (1.2496206)/((Cos(phi))+(1.2499299))
Generation 1676, Farm 41, MDD 35, Fitness 0.04000006727012346: (1.2496226)/((Cos(phi))+(1.2499299))
Generation 1679, Farm 41, MDD 38, Fitness 0.04000005719847313: (1.2496206)/((Cos(phi))+(1.2499229))
Generation 1681, Farm 41, MDD 40, Fitness 0.04000005680320688: (1.2496206)/((Cos(phi))+(1.2499226))
Generation 1683, Farm 41, MDD 42, Fitness 0.04000005541298644: (1.2496406)/((Cos(phi))+(1.2499299))
Generation 1684, Farm 41, MDD 43, Fitness 0.04000003252961008: (1.2496906)/((Cos(phi))+(1.2499299))
```

```
Generation 1685, Farm 41, MDD 44, Fitness 0.04000002593819688: (1.2497208)/((Cos(phi))+(1.2499296))
Generation 1689, Farm 41, MDD 48, Fitness 0.04000002586953529: (1.2497208)/((Cos(phi))+(1.2499299))
Generation 1691, Farm 41, MDD 50, Fitness 0.04000000649574147: (1.2499206)/((Cos(phi))+(1.2499708))
Generation 1694, Farm 41, MDD 53, Fitness 0.04000000210653838: (1.2499205)/((Cos(phi))+(1.2499799))
Generation 1699, Farm 41, MDD 58, Fitness 0.04000000203874848: (1.2499296)/((Cos(phi))+(1.2499799))
Generation 1701, Farm 41, MDD 60, Fitness 0.04000000202093067: (1.2499285)/((Cos(phi))+(1.2499799))
Generation 1706, Farm 41, MDD 55, Fitness 0.04000000201643962: (1.2499245)/((Cos(phi))+(1.2499799))
Generation 1709, Farm 41, MDD 52, Fitness 0.04000000201094011: (1.2499275)/((Cos(phi))+(1.2499799))
Generation 1713, Farm 41, MDD 48, Fitness 0.04000000129155995: (1.2499999)/((Cos(phi))+(1.2499946))
Generation 1714, Farm 41, MDD 47, Fitness 0.04000000115097394: (1.2499799)/((Cos(phi))+(1.2499999))
Generation 1715, Farm 41, MDD 46, Fitness 0.04000000000025976: (1.2499999)/((Cos(phi))+(1.2499999))
Generation 1730, Farm 41, MDD 31, Fitness 0.04000000000010225: (1.2499995)/((Cos(phi))+(1.2499999))
Generation 1732, Farm 41, MDD 29, Fitness 0.04000000000005297: (1.2499996)/((Cos(phi))+(1.2499999))
Generation 6468, Farm 6, MDD 27, Fitness 0.04000000000002955: (1.2499999)/((Cos(phi))+(1.25))
Generation 6488, Farm 6, MDD 47, Fitness 0.04: (1.25)/((Cos(phi))+(1.25))
```

13.4.7 Round 2, Simulation 4

```
Generation 1, Farm 45, MDD 60, Fitness 4.46610681350701:
   (Sqrt(Cosh(Cos((-phi)))))-(ArcTan(6.326482)):((Sqrt(Abs(phi)))+(Log((-Sin(4.421339)))))/(66481.409)
Generation 2, Farm 27, MDD 61, Fitness 4.099037843896236:
   Cos((-phi)):(Tanh((585421.67)^((-((-(-5020.3289)))+(2)))))+(Tanh(phi))
Generation 7, Farm 27, MDD 66, Fitness 3.148692948137608:
   Cos((-phi)):(Tanh((585421.67)^((-((-(-5020.3289)))+(2)))))*(Tanh(phi))
Generation 9, Farm 27, MDD 68, Fitness 3.148692948137608:
   Cos((-phi)):(Tanh((585421.67)^((-((-(-5020.3285)))+(2)))))*(Tanh(phi))
Generation 12, Farm 27, MDD 71, Fitness 3.148692948137608:
   Cos((-phi)):(Tanh((585421.67)^((-((-(-5000.3285)))+(2)))))*(Tanh(phi))
Generation 13, Farm 8, MDD 72, Fitness 2.552730742489057:
   (-(Pi()))*((Cos(Sinh(((-1.204771))-(3.29935))))^((289.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 15, Farm 8, MDD 74, Fitness 2.552411181072483:
   (-(Pi()))*((Cos(Sinh(((-1.204771))-(3.29935))))^((288.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 16, Farm 27, MDD 75, Fitness 2.415693629602848:
   Cos((-phi)):(Tanh((Abs((-1.4710051)))^((-((-(-5020.3289)))+(2)))))+(Sin(phi))
Generation 17, Farm 8, MDD 76, Fitness 1.773351458502076:
   (-(Pi()))*((Exp(Sinh(((-1.204771))-(3.29935))))^((289.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 21, Farm 8, MDD 80, Fitness 1.771792990814803:
   (-(Pi()))*((Exp(Sinh(((-1.204771))-(3.29535))))^((289.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 22, Farm 14, MDD 81, Fitness 1.771017207770848:
   (-(Pi()))*((Exp(Sinh(((-1.202771))-(3.29535))))^((289.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 24, Farm 8, MDD 83, Fitness 1.626909436151734:
   (-(Pi()))*((Exp(Sinh(((-1.204751))-(3.29935))))^((2180.8)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 30, Farm 14, MDD 89, Fitness 1.599770471918878:
   (-(Pi()))*((Exp(Sinh(((-1.202773))-(3.29539))))^((989.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 31, Farm 14, MDD 90, Fitness 1.599770437837861:
   (-(Pi()))*((Exp(Sinh(((-1.202777))-(3.29539))))^((989.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 32, Farm 14, MDD 91, Fitness 1.374151837964201:
   (-(4.862091)*((Exp(Sinh(((-1.202771))-(3.29535))))^((289.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 38, Farm 14, MDD 97, Fitness 1.366597217804409:
   (-(4.862091)*((Exp(Sinh(((-1.272771))-(3.29535))))^((289.94398)^(Cos(phi)))):ArcCos(Tanh(phi))
Generation 43, Farm 20, MDD 102, Fitness 1.015114836786049:
   Tan((4.231)+((Cos((phi)-(Exp(Log(119.1165)))))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tanh(Cos((phi)-(Exp(Log(8.223333)))))
Generation 49, Farm 20, MDD 108, Fitness 0.9535351793772079:
   Tan((4.231)+((Cos((phi)-(Exp(Log(119.1165)))))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tanh(Cos((phi)-(Exp(Log(8.023333)))))
Generation 51, Farm 20, MDD 110, Fitness 0.7015783137922128:
   Tan((4.231)+((Cos((phi)-(Exp(Log(119.3165)))))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tanh(Cos((phi)-(Exp(Log(8.223333)))))
Generation 53, Farm 20, MDD 112, Fitness 0.6812960522102486:
   Tan((4.231)+((Cos((phi)-(Exp(Log(3.5580062E-89)))))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tanh(Cos((phi)-(Exp(Log(8.223333)))))
Generation 60, Farm 20, MDD 119, Fitness 0.6562960522102486:
   Tan((4.231)+((Cos(phi))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tanh(Cos((phi)-(Exp(Log(8.223333)))))
Generation 61, Farm 20, MDD 120, Fitness 0.5947163948014069:
   Tan((4.231)+((Cos(phi))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tanh(Cos((phi)-(Exp(Log(8.023333)))))
Generation 70, Farm 20, MDD 111, Fitness 0.5842289191953525:
   Tan((4.231)+((Cos(phi))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tan(Cos((phi)-(Exp(Log(8.023333)))))
Generation 72, Farm 20, MDD 109, Fitness 0.5827247273750544:
   Tan((4.231)+((Cos(phi))+((4.2)-(Sin((Sinh((-119.1165)))+(ArcSinh(phi)))))):Tanh(Cos((phi)-(ArcSinh(Log(8.023333)))))
Generation 76, Farm 49, MDD 105, Fitness 0.4289284655994218:
   Sinh((ArcSin(Cos(phi)))+((-ArcSinh(Tanh(Abs(Sinh(Cos(phi))))))):Sin(phi)
Generation 80, Farm 49, MDD 101, Fitness 0.4254254994931061:
   Sinh((ArcSin(Cos(phi)))+((-ArcSinh(Sin(Abs(Sinh(Cos(phi))))))):Sin(phi)
Generation 83, Farm 4, MDD 98, Fitness 0.4013191328395023:
   Tan((Cos(phi))+(5.93)):Sin(phi)
Generation 90, Farm 4, MDD 91, Fitness 0.3912360238188002:
   Tan((Cos(phi))-(9.7988105)):Sin(phi)
Generation 98, Farm 4, MDD 83, Fitness 0.3908028328169936:
   Tan((Cos(phi))-(9.7982105)):Sin(phi)
Generation 99, Farm 4, MDD 82, Fitness 0.3895643357854336:
   Tan((Cos(phi))-(9.7958105)):Sin(phi)
Generation 100, Farm 4, MDD 81, Fitness 0.3891440387244045:
   Tan((Cos(phi))-(9.7919105)):Sin(phi)
Generation 108, Farm 4, MDD 73, Fitness 0.3891429872872677:
   Tan((Cos(phi))-(9.7919185)):Sin(phi)
```

```
Generation 109, Farm 4, MDD 72, Fitness 0.3890722495518118:
   Tan((Cos(phi))-(9.7931105)):Sin(phi)
Generation 111, Farm 4, MDD 70, Fitness 0.3664294318233745:
   (Cos(phi))/(Exp(Tan(Cos(phi)))):Sin(phi)
Generation 115, Farm 23, MDD 66, Fitness 0.3497316713305653:
   ((Cos(phi))/(Cos(Cos(phi))))/(Exp(Cos((-phi)))):(Sin(phi))/(Cos(Cos(phi)))
Generation 120, Farm 23, MDD 61, Fitness 0.3397316713305653:
   ((Cos(phi))/(Cos(Cos(phi))))/(Exp(Cos(phi))):(Sin(phi))/(Cos(Cos(phi)))
Generation 127, Farm 23, MDD 54, Fitness 0.1671789657635615:
   ((Cos(phi))/(Cos(Cos(phi))))/(Exp(Cos((-phi)))):(Sin(phi))/(Exp(Cos(phi)))
Generation 135, Farm 23, MDD 46, Fitness 0.1571789657635616:
   ((Cos(phi))/(Cos(Cos(phi))))/(Exp(Cos(phi))):(Sin(phi))/(Exp(Cos(phi)))
Generation 145, Farm 23, MDD 44, Fitness 0.1571789657635615:
   ((Cos(phi))/(Exp(Cos(phi))))/(Cos(Cos(phi))):(Sin(phi))/(Exp(Cos(phi)))
Generation 148, Farm 23, MDD 47, Fitness 0.1549719561425042:
   ((Cos(phi))/(Cos(Cos(phi))))/(Exp(Cos(phi))):(Sin((Sin(phi))/(Cos(Cos(phi)))))/(Exp(Cos(phi)))
Generation 152, Farm 4, MDD 51, Fitness 0.1117046305624986:
   (Cos(phi))/(Exp(Tan(Cos(phi)))):(Sin(phi))/(Exp(Tan(Cos(phi))))
Generation 259, Farm 4, MDD 82, Fitness 0.1063624229864468:
   (Cos(phi))/(ArcTan(Exp(Tan(Cos(phi))))):(Sin(phi))/(Exp(Tan(Cos(phi))))
Generation 267, Farm 4, MDD 74, Fitness 0.09705118039912053:
   (Cos(phi))/(ArcSinh(Exp(Tan(Cos(phi))))):(Sin(phi))/(Exp(Tan(Cos(phi))))
```

13.4.8 Round 3, Simulation 1

```
Generation 1, Farm 133, MDD 80, Fitness 5741498.8262467654:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000797951E-08),(ArcSinh(&G000000007760880278E-08))-(Cosh(&G0000000000000054893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00707058),(ArcSinh(72.60880271000001))-(Cosh(0.00055103)))))

Generation 6, Farm 133, MDD 85, Fitness 5740651.4038857194:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000797901E-08),(ArcSinh(&G000000007760880278E-08))-(Cosh(&G0000000000000054893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00707001),(ArcSinh(72.60880271000001))-(Cosh(0.00055103)))))

Generation 10, Farm 133, MDD 89, Fitness 4132912.8604141511:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000597951E-08),(ArcSinh(&G000000007760880278E-08))-(Cosh(&G0000000000000054893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00507058),(ArcSinh(72.60880271000001))-(Cosh(0.00055103)))))

Generation 12, Farm 133, MDD 91, Fitness 4132912.8604141511:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000597951E-08),(ArcSinh(&G000000007760880278E-08))-(Cosh(&G0000000000000654893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00507058),(ArcSinh(72.60880271000001))-(Cosh(0.00655103)))))

Generation 14, Farm 36, MDD 93, Fitness 4053010.0997015568:
((Log(Cosh((Cos(Max(ArcTan((-&G000000000000968837E-08)),(-&G000000000448626113E-08))))*(t))))*(Pi()))*((-&G000000005177653195E-08))
((Log(Cosh((Cos(Max(ArcTan((-0.009311669999999999)),(-4.48626183))))*(t))))*(Pi()))*((-58.72656105))

Generation 17, Farm 89, MDD 96, Fitness 3148567.9625028558:
(-((Abs((-t)))^(ArcSinh(&G000000000874900692E-08)))*(Tan(&G000007405107573989E-08)))
(-((Abs((-t)))^(ArcSinh(8.750006969999999)))*(Tan(75941.92523089)))

Generation 20, Farm 199, MDD 99, Fitness 2158793.0601854795:
((-t))*(((Cos(&G000000000000008934E-08))^((-&G040000000000008199E-08)))*(t))
((-t))*(((Cos(8.964E-05))^((-400000000.0000811)))*(t))

Generation 28, Farm 199, MDD 93, Fitness 2158793.0601848541:
((-t))*(((Cos(&G000000000000008934E-08))^((-&G040000000000000199E-08)))*(t))
((-t))*(((Cos(8.964E-05))^((-400000000.0000011)))*(t))

Generation 29, Farm 199, MDD 92, Fitness 2152854.189275268:
((-t))*(((Cos(&G000000000000008944E-08))^((-&G040000000000800001E-08)))*(t))
((-t))*(((Cos(8.954999999999999E-05))^((-400000000.008)))*(t))

Generation 33, Farm 119, MDD 88, Fitness 2142365.3440660075:
((ArcSinh(Tan((&G009702046005970279E-08)-(&G400148502764074863E-08))))*(t))*((t)+(t))
((ArcSinh(Tan((92020460.05079728)-(4001514027.359249))))*(t))*((t)+(t))

Generation 37, Farm 158, MDD 84, Fitness 2130371.0999889527:
(-Abs(((-((-Abs(((-Exp(ArcCos((-&G000000000000000728E-08)))))*(t))))+(Tanh(&G000000000330668945E-08))))*(t)))
(-Abs(((-((-Abs(((-Exp(ArcCos((-7.710000000000001E-06)))))*(t))))+(Tanh(3.60668954))))*(t)))

Generation 40, Farm 105, MDD 81, Fitness 1011345.8990925616:
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(&G000000810000000069E-08))))))-(Min(t,(ArcSinh(&G070000526809161264E-08))/(Pi()))))
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(8199.9999993))))))-(Min(t,(ArcSinh(799994268.0986173))/(Pi()))))

Generation 47, Farm 105, MDD 74, Fitness 1011345.8977101628:
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(&G000000810000000069E-08))))))-(Min(t,(ArcSinh(&G070000566809161264E-08))/(Pi()))))
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(8199.9999993))))))-(Min(t,(ArcSinh(799994668.0986173))/(Pi()))))

Generation 49, Farm 105, MDD 72, Fitness 993540.60222571553:
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(&G000000000000000009E-08)))))-(Min(t,t)))
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(9E-08)))))-(Min(t,t)))
```

```
Generation 50, Farm 105, MDD 71, Fitness 929703.30783702992:
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(&G000000810000000069E-08))))))-(Min(t,(t)/(Pi()))))
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(8199.9999993))))))-(Min(t,(t)/(Pi()))))

Generation 54, Farm 105, MDD 67, Fitness 929702.17856321111:
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(&G000009810000000069E-08))))))-(Min(t,(t)/(Pi()))))
(t)*((((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(91800.00000068999))))))-(Min(t,(t)/(Pi()))))

Generation 55, Farm 158, MDD 66, Fitness 861037.99228423636:
(-Abs((Abs(((-Log(Max(t,&G0000000000000073856E-08))))*(t)))*(t)))
(-Abs((Abs(((-Log(Max(t,0.00076853))))*(t)))*(t)))

Generation 60, Farm 105, MDD 61, Fitness 371530.45462239615:
(t)*((((-t))+((((-t))+((-t)))-(Min(t,(t)-((-t))))))-(Min(t,(ArcSinh(&G0700005268091612 64E-08))/(Pi()))))
(t)*((((-t))+((((-t))+((-t)))-(Min(t,(t)-((-t))))))-(Min(t,(ArcSinh(799994268.0986173))/(Pi()))))

Generation 62, Farm 105, MDD 59, Fitness 344250.95476854627:
(t)*((((-t))+(((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(&G030006810000400069E-08)))))))-(Min(t,Pi())))
(t)*((((-t))+(((-t))+((((-t))+((-t)))-(Min(t,Sinh(ArcTan(399931800.0040007)))))))-(Min(t,Pi())))

Generation 67, Farm 105, MDD 54, Fitness 145277.57966533853:
(t)*((((-t))+((((-t))+((-t)))-(Min(t,(t)-((-t))))))-(Min(t,((-t))/(Pi()))))
(t)*((((-t))+((((-t))+((-t)))-(Min(t,(t)-((-t))))))-(Min(t,((-t))/(Pi()))))

Generation 69, Farm 63, MDD 52, Fitness 133060.83933405369:
(-((t)*(ArcSinh((-(&G0000000000000398176E-08)^(t)))))-((-(t)*(ArcSinh((-(t)^(t)))))))
(-((t)*(ArcSinh((-(0.00308126)^(t)))))-((-(t)*(ArcSinh((-(t)^(t)))))))

Generation 70, Farm 63, MDD 51, Fitness 122399.45371343223:
((t)*(ArcSinh((-(-(&G0000000000000398176E-08)^(t))))))-((-Min((t)*(ArcSinh((-(t)^(t)))),t)))
((t)*(ArcSinh((-(-(0.00308126)^(t))))))-((-Min((t)*(ArcSinh((-(t)^(t)))),t)))

Generation 76, Farm 158, MDD 45, Fitness 92916.713746166468:
(-Abs((Abs(((-Log(Max(t,ArcSinh(&G000000000000200728E-08)))))*(t)))*(t)))
(-Abs((Abs(((-Log(Max(t,ArcSinh(0.00200771)))))*(t)))*(t)))

Generation 100, Farm 158, MDD 39, Fitness 92742.551003524364:
(-Abs((Abs(((-Log(Max(t,ArcSinh(&G000000000000200928E-08)))))*(t)))*(t)))
(-Abs((Abs(((-Log(Max(t,ArcSinh(0.00200971)))))*(t)))*(t)))

Generation 106, Farm 158, MDD 45, Fitness 87039.173085910559:
(-Abs((Abs(((-Log(Max(t,ArcSinh(&G000000000000208718E-08)))))*(t)))*(t)))
(-Abs((Abs(((-Log(Max(t,ArcSinh(0.00208788)))))*(t)))*(t)))

Generation 116, Farm 158, MDD 55, Fitness 86784.206521698987:
(-Abs((Abs(((-Log(Max(t,Abs(&G000000000000209720E-08)))))*(t)))*(t)))
(-Abs((Abs(((-Log(Max(t,Abs(0.0020922)))))*(t)))*(t)))

Generation 117, Farm 158, MDD 56, Fitness 84774.397672354346:
(-Abs((Abs(((-Log(Max(t,&G000000000000247128E-08))))*(t)))*(t)))
(-Abs((Abs(((-Log(Max(t,0.00247828))))*(t)))*(t)))

Generation 118, Farm 158, MDD 57, Fitness 81076.70291573026:
(-Abs((Abs(((-Log(Max(t,ArcSinh(&G000000000000228718E-08)))))*(t)))*(t)))
(-Abs((Abs(((-Log(Max(t,ArcSinh(0.00228788)))))*(t)))*(t)))

Generation 120, Farm 175, MDD 59, Fitness 9190.3208309014499:
((-t))*(((-Tan(Sqrt(Pi())))*((-(&G000000000900000630E-08)+(t))))
((-t))*(((-Tan(Sqrt(Pi())))*((-(9.9999936)+(t))))

Generation 125, Farm 175, MDD 64, Fitness 9190.3173120900701:
((-t))*(((-Tan(Sqrt(Pi())))*((-(&G000000000900000600E-08)+(t))))
((-t))*(((-Tan(Sqrt(Pi())))*((-(9.99999399)+(t))))

Generation 127, Farm 175, MDD 66, Fitness 9019.638161019273:
((-t))*(((-Tan(Sqrt(Pi())))*((-(&G000000001901020562E-08)+(t))))
((-t))*(((-Tan(Sqrt(Pi())))*((-(10.01979462)+(t))))

Generation 128, Farm 175, MDD 67, Fitness 8861.9496550770382:
((-t))*(((-Tan(Sqrt(Pi())))*((-(&G000000001903000562E-08)+(t))))
((-t))*(((-Tan(Sqrt(Pi())))*((-(10.03999462)+(t))))

Generation 135, Farm 175, MDD 74, Fitness 8598.6065036909331:
((-t))*(((-Tan(Sqrt(Pi())))*((-(&G000000001908000562E-08)+(t))))
((-t))*(((-Tan(Sqrt(Pi())))*((-(10.08000537)+(t))))

Generation 136, Farm 56, MDD 75, Fitness 6538.0379538159905:
((t)*(((-((-Pi()))*(Pi())))*(&G000000000490105411E-08)))+((t)*(((-t))*(&G000000000480105410E-08)))
((t)*(((-((-Pi()))*(Pi())))*(4.99805581)))+((t)*(((-t))*(4.80194419)))

Generation 141, Farm 178, MDD 80, Fitness 4619.3091557348043:
(&G000000000484809039E-08)*((t)-((t)*((t)-(&G000000000964869039E-08))))
(4.84809969)*((t)-((t)*((t)-(9.3513003))))
```

Generation 147, Farm 178, MDD 86, Fitness 4471.415033670457:
(&G000000000484009039E-08)*((t)-((t)*((t)-(&G00000000964869039E-08))))
(4.84009969)*((t)-((t)*((t)-(9.3513003))))

Generation 151, Farm 178, MDD 90, Fitness 4470.9214748069362:
(&G000000000484009039E-08)*((t)-((t)*((t)-(&G00000000964969039E-08))))
(4.84009969)*((t)-((t)*((t)-(9.350699690000001))))

Generation 162, Farm 178, MDD 99, Fitness 4470.4427615644272:
(&G000000000484009039E-08)*((t)-((t)*((t)-(&G00000000964909039E-08))))
(4.84009969)*((t)-((t)*((t)-(9.35009969))))

Generation 163, Farm 178, MDD 98, Fitness 4469.3816725562083:
(&G000000000484009039E-08)*((t)-((t)*((t)-(&G00000000965869009E-08))))
(4.84009969)*((t)-((t)*((t)-(9.3486999))))

Generation 173, Farm 178, MDD 88, Fitness 4468.3986039588845:
(&G000000000484009039E-08)*((t)-((t)*((t)-(&G00000000965769009E-08))))
(4.84009969)*((t)-((t)*((t)-(9.347300089999999))))

Generation 174, Farm 178, MDD 87, Fitness 4462.6904478631395:
(&G000000000484009039E-08)*((t)-((t)*((t)-(&G00000000966969039E-08))))
(4.84009969)*((t)-((t)*((t)-(9.330699689999999))))

Generation 178, Farm 88, MDD 83, Fitness 3973.8590071999706:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245709E-08))))-((ArcCosh(&G00064000630364960 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245209))))-((ArcCosh(6400063.9664939))*(Pi()))))-((t)*(t))

Generation 187, Farm 140, MDD 74, Fitness 3220.6557033929771:
(-(Abs((-((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))*(t)))))*(ArcTan(t)))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))))
(-(Abs((-((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))*(t)))))*(ArcTan(t)))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))))

Generation 197, Farm 140, MDD 64, Fitness 3218.6557033929771:
(-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))*(t)))))*(ArcTan(t))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))))
(-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))*(t)))))*(ArcTan(t))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))))

Generation 224, Farm 88, MDD 37, Fitness 3116.3206219774147:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245709E-08))))-((ArcCosh(&G00054000620364960 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245209))))-((ArcCosh(5599937.9664939))*(Pi()))))-((t)*(t))

Generation 225, Farm 88, MDD 36, Fitness 2412.0591582307225:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245709E-08))))-((ArcCosh(&G00034000620364960 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245209))))-((ArcCosh(3599937.9664939))*(Pi()))))-((t)*(t))

Generation 241, Farm 88, MDD 40, Fitness 2285.7430671907964:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245709E-08))))-((ArcCosh(&G00042000620364260 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245209))))-((ArcCosh(4200062.0335739))*(Pi()))))-((t)*(t))

Generation 257, Farm 88, MDD 56, Fitness 2285.7430670612016:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245709E-08))))-((ArcCosh(&G00042000620374260 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245209))))-((ArcCosh(4200062.03242609))*(Pi()))))-((t)*(t))

Generation 259, Farm 88, MDD 58, Fitness 2285.7430668353968:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245709E-08))))-((ArcCosh(&G00042000620394260 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245209))))-((ArcCosh(4200062.03042609))*(Pi()))))-((t)*(t))

Generation 265, Farm 88, MDD 64, Fitness 2285.7428428009689:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245704E-08))))-((ArcCosh(&G00042000620363260 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245204))))-((ArcCosh(4200062.03362609))*(Pi()))))-((t)*(t))

Generation 268, Farm 88, MDD 67, Fitness 2282.7481779803693:
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303145709E-08))))-((ArcCosh(&G00042000620394260 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((ArcCosh(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.9615479))))-((ArcCosh(4200062.03042609))*(Pi()))))-((t)*(t))

Generation 270, Farm 88, MDD 69, Fitness 2231.7463539246328:
(((-((-ArcSinh((Abs(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245709E-08))))-((ArcCosh(&G00042000620364260 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((Abs(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245209))))-((ArcCosh(4200062.0335739))*(Pi()))))-((t)*(t))

Generation 275, Farm 88, MDD 74, Fitness 2231.7401629424735:
(((-((-ArcSinh((Abs(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-&G000000000303245729E-08))))-((ArcCosh(&G00042000620364260 9E-08))*(Pi()))))-((t)*(t))
(((-((-ArcSinh((Abs(Pi()))*(t))))*(t)))-((t)*(((-(t)*((-3.96245229))))-((ArcCosh(4200062.0335739))*(Pi()))))-((t)*(t))

Generation 279, Farm 140, MDD 78, Fitness 2168.4404740874184:
(-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*(((Cos(Pi()))*(Pi()))-(t)))))*(ArcTan(t))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))))
(-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*(((Cos(Pi()))*(Pi()))-(t)))))*(ArcTan(t))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))))

Generation 280, Farm 140, MDD 79, Fitness 1928.0526383812341:
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))-(t)))))*(ArcTan(t))))+(((-(-(Pi())*(Pi())))))*(ArcTan(Pi()))))+
(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi()))))
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))-(t)))))*(ArcTan(t))))+(((-(-(Pi())*(Pi())))))*(ArcTan(Pi()))))+
(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi()))))

Generation 286, Farm 140, MDD 85, Fitness 1008.2140804500434:
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))-(t)))))*(ArcTan(t))))+(((-(-(Pi()))^(Pi())))*(ArcTan(Pi()))))+
(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))-(t)))))*(ArcTan(t))))+(((-(-(Pi()))^(Pi())))*(ArcTan(Pi()))))+
(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))

Generation 301, Farm 140, MDD 100, Fitness 976.52979804354698:
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))*(t)))))*(ArcTan(t))))+(((-(-(Pi()))^(Pi())))*(ArcTan(Pi()))))+
(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos(Pi()))*(t)))))*(ArcTan(t))))+(((-(-(Pi()))^(Pi())))*(ArcTan(Pi()))))+
(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))

Generation 313, Farm 179, MDD 88, Fitness 652.08524260130173:
(((-(t)*(((t)+(t))+((-&G000000004914683452E-08)))))+(((-(t)*((t)+((t)+(t)))))+((Sinh(Pi()))+(&G000000008679673219E-08))))+((t)*(ArcSinh(t)))
(((-(t)*(((t)+(t))+((-49.84683542)))))+(((-(t)*((t)+((t)+(t)))))+((Sinh(Pi()))+(86.70676210000001))))+((t)*(ArcSinh(t)))

Generation 316, Farm 140, MDD 85, Fitness 646.96472472001449:
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos((Abs(Pi()))*(Pi())))*(t)))))*(ArcTan(t))))+
(((-(-(Pi()))^(Pi())))))*(ArcTan(Pi())))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))
((-(Abs(((-(Abs(((Pi())*(t))*(t)))-((Pi())*((Cos((Abs(Pi()))*(Pi())))*(t)))))*(ArcTan(t))))+
(((-(-(Pi()))^(Pi())))))*(ArcTan(Pi())))))+(((-(-((Sinh(Pi()))*(t))*(Pi())))))*(ArcTan(Pi())))

Generation 328, Farm 212, MDD 73, Fitness 328.65217071388827:
(&G000000007000420040E-08)+((t)*((&G000000005903603040E-08)+((&G000000000488006582E-08)*((-t)))))
(79.99579959)+((t)*((50.0339604)+((4.88006517)*((-t)))))

Generation 340, Farm 212, MDD 61, Fitness 328.54625644401233:
(&G000000007000420040E-08)+((t)*((&G000000005903603040E-08)+((&G000000000488006982E-08)*((-t)))))
(79.99579959)+((t)*((50.0339604)+((4.88006917)*((-t)))))

Generation 344, Farm 212, MDD 57, Fitness 328.54021402264715:
(&G000000007000420040E-08)+((t)*((&G000000005903803240E-08)+((&G000000000488006582E-08)*((-t)))))
(79.99579959)+((t)*((50.0319624)+((4.88006517)*((-t)))))

Generation 345, Farm 215, MDD 56, Fitness 63.998257046024243:
(((((-t))*(&G000000000489200864E-08))*(t))+(&G000000009903286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.89799135))*(t))+(90.03713549))+((t)*(49.99009659))

Generation 354, Farm 215, MDD 47, Fitness 63.565516085224814:
(((((-t))*(&G000000000489200864E-08))*(t))+(&G000000009907286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.89799135))*(t))+(90.07713549))+((t)*(49.99009659))

Generation 360, Farm 215, MDD 41, Fitness 62.653278360394637:
(((((-t))*(&G000000000489250864E-08))*(t))+(&G000000009903286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.89740864))*(t))+(90.03713549))+((t)*(49.99009659))

Generation 369, Farm 215, MDD 32, Fitness 62.395525303505138:
(((((-t))*(&G000000000489270894E-08))*(t))+(&G000000009903286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.89720895))*(t))+(90.03713549))+((t)*(49.99009659))

Generation 370, Farm 215, MDD 31, Fitness 61.875242931835544:
(((((-t))*(&G000000000489290864E-08))*(t))+(&G000000009907286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.89700864))*(t))+(90.07713549))+((t)*(49.99009659))

Generation 373, Farm 215, MDD 32, Fitness 34.847112926022831:
(((((-t))*(&G000000000489200864E-08))*(t))+(&G000000009303286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.89799135))*(t))+(96.03713549))+((t)*(49.99009659))

Generation 375, Farm 215, MDD 34, Fitness 34.698749466523864:
(((((-t))*(&G000000000489200864E-08))*(t))+(&G000000009403286459E-08))+((t)*(&G000000004900409350E-08))
(((((-t))*(4.89799135))*(t))+(95.96286449999999))+((t)*(49.9959034))

Generation 384, Farm 215, MDD 43, Fitness 34.698381444461901:
(((((-t))*(&G000000000489200864E-08))*(t))+(&G000000009403286459E-08))+((t)*(&G000000004900404350E-08))
(((((-t))*(4.89799135))*(t))+(95.96286449999999))+((t)*(49.99595659))

Generation 385, Farm 215, MDD 44, Fitness 34.660632106526272:
(((((-t))*(&G000000000489200864E-08))*(t))+(&G000000009407286459E-08))+((t)*(&G000000004900409350E-08))
(((((-t))*(4.89799135))*(t))+(95.9228645))+((t)*(49.9959034))

Generation 394, Farm 215, MDD 53, Fitness 27.172731839634899:
(((((-t))*(&G000000000499200864E-08))*(t))+(&G000000009303286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.90200864))*(t))+(96.03713549))+((t)*(49.99009659))

Generation 404, Farm 215, MDD 63, Fitness 19.858676159344565:
(((((-t))*(&G000000000499200864E-08))*(t))+(&G000000009203286459E-08))+((t)*(&G000000004900409350E-08))
(((((-t))*(4.90200864))*(t))+(97.96286449999999))+((t)*(49.9959034))

Generation 417, Farm 215, MDD 76, Fitness 19.855007241344587:
(((((-t))*(&G000000000499200864E-08))*(t))+(&G000000009203486459E-08))+((t)*(&G000000004900409350E-08))
(((((-t))*(4.90200864))*(t))+(97.9648645))+((t)*(49.9959034))

Generation 420, Farm 215, MDD 79, Fitness 19.844419279344574:
(((((-t))*(&G0000000000499200864E-08))*(t))+(&G000000009203286459E-08))+((t)*(&G000000004900009350E-08))
(((((-t))*(4.90200864))*(t))+(97.96286449999999))+((t)*(49.9999034))

Generation 423, Farm 215, MDD 82, Fitness 19.159303975281365:
(((((-t))*(&G0000000000499230864E-08))*(t))+(&G000000009903286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.90239135))*(t))+(99.96286449999999))+((t)*(49.99009659))

Generation 428, Farm 215, MDD 87, Fitness 18.611543575282653:
(((((-t))*(&G0000000000499230864E-08))*(t))+(&G000000009063286459E-08))+((t)*(&G000000004900909350E-08))
(((((-t))*(4.90239135))*(t))+(99.3628645))+((t)*(49.99009659))

Generation 434, Farm 215, MDD 93, Fitness 18.57219426336566:
(((((-t))*(&G0000000000499260864E-08))*(t))+(&G000000009009286419E-08))+((t)*(&G000000004900402350E-08))
(((((-t))*(4.90260864))*(t))+(99.9028641))+((t)*(49.99597659))

Generation 437, Farm 215, MDD 96, Fitness 18.139336171525052:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009003286459E-08))+((t)*(&G000000004900409350E-08))
(((((-t))*(4.90299135))*(t))+(99.96286449999999))+((t)*(49.9959034))

Generation 440, Farm 215, MDD 99, Fitness 18.139332062931206:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009003286459E-08))+((t)*(&G000000004900409300E-08))
(((((-t))*(4.90299135))*(t))+(99.96286449999999))+((t)*(49.99590399))

Generation 444, Farm 215, MDD 97, Fitness 18.125079291525115:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009003286459E-08))+((t)*(&G000000004900009350E-08))
(((((-t))*(4.90299135))*(t))+(99.96286449999999))+((t)*(49.9999034))

Generation 446, Farm 215, MDD 95, Fitness 18.125079194931171:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009003286459E-08))+((t)*(&G000000004900009300E-08))
(((((-t))*(4.90299135))*(t))+(99.96286449999999))+((t)*(49.99990399))

Generation 449, Farm 215, MDD 92, Fitness 18.112856192343198:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009009286419E-08))+((t)*(&G000000004900402330E-08))
(((((-t))*(4.90299135))*(t))+(99.9028641))+((t)*(49.99597639))

Generation 453, Farm 215, MDD 88, Fitness 18.112833819951106:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009009280419E-08))+((t)*(&G000000004900402330E-08))
(((((-t))*(4.90299135))*(t))+(99.9028041))+((t)*(49.99597639))

Generation 454, Farm 215, MDD 87, Fitness 18.112575368724315:
(((((-t))*(&G0000000000499290824E-08))*(t))+(&G000000009009286419E-08))+((t)*(&G000000004900402330E-08))
(((((-t))*(4.90299175))*(t))+(99.9028641))+((t)*(49.99597639))

Generation 455, Fitness 18, MDD 86, Fitness 18.107593745511018:
(((((-t))*(&G0000000000499290064E-08))*(t))+(&G000000009009286419E-08))+((t)*(&G000000004900406330E-08))
(((((-t))*(4.90299935))*(t))+(99.9028641))+((t)*(49.99593639))

Generation 462, Farm 215, MDD 79, Fitness 18.107396140438105:
(((((-t))*(&G0000000000499290034E-08))*(t))+(&G000000009009286419E-08))+((t)*(&G000000004900406330E-08))
(((((-t))*(4.90299964))*(t))+(99.9028641))+((t)*(49.99593639))

Generation 463, Farm 215, MDD 78, Fitness 18.106879079110872:
(((((-t))*(&G0000000000499290064E-08))*(t))+(&G000000009009086419E-08))+((t)*(&G000000004900406330E-08))
(((((-t))*(4.90299935))*(t))+(99.90086410000001))+((t)*(49.99593639))

Generation 465, Farm 215, MDD 76, Fitness 18.070615775665452:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009033286459E-08))+((t)*(&G000000004900009350E-08))
(((((-t))*(4.90299135))*(t))+(99.63713549000001))+((t)*(49.9999034))

Generation 480, Farm 215, MDD 61, Fitness 18.070500741525269:
(((((-t))*(&G0000000000499290864E-08))*(t))+(&G000000009033216459E-08))+((t)*(&G000000004900009350E-08))
(((((-t))*(4.90299135))*(t))+(99.63786450000001))+((t)*(49.9999034))

Generation 482, Farm 215, MDD 59, Fitness 18.070493176008977:
(((((-t))*(&G0000000000499290814E-08))*(t))+(&G000000009033286459E-08))+((t)*(&G000000004900009350E-08))
(((((-t))*(4.90299184))*(t))+(99.63713549000001))+((t)*(49.9999034))

Generation 485, Farm 215, MDD 56, Fitness 18.070488553903377:
(((((-t))*(&G0000000000499290814E-08))*(t))+(&G000000009033283459E-08))+((t)*(&G000000004900009350E-08))
(((((-t))*(4.90299184))*(t))+(99.6371645))+((t)*(49.9999034))

Generation 486, Farm 14, MDD 55, Fitness 16.373854245034373:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901324007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01624007))*(t))+(99.92182004999999)

Generation 493, Farm 14, MDD 48, Fitness 16.373753880251979:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901324007E-08))*(t))+(&G000000009007132005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01624007))*(t))+(99.92162005)

Generation 500, Farm 14, MDD 41, Fitness 16.373726399869607:
(((((-t))*(ArcSinh(&G000000006766177907E-08)))+(&G000000005901324007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.33872992000001)))+(50.01624007))*(t))+(99.92182004999999)

Generation 503, Farm 14, MDD 38, Fitness 16.371646914715608:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901314007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01615992))*(t))+(99.92182004999999)

Generation 506, Farm 14, MDD 35, Fitness 16.277021769033929:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901724007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01224007))*(t))+(99.92182004999999)

Generation 507, Farm 14, MDD 34, Fitness 16.238705166251954:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009007132005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01024007))*(t))+(99.92162005)

Generation 513, Farm 14, MDD 32, Fitness 16.214069111175263:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901724007E-08))*(t))+(&G000000009037112005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01224007))*(t))+(99.67817994000001)

Generation 516, Farm 14, MDD 35, Fitness 16.214063934588946:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901724007E-08))*(t))+(&G000000009037212005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01224007))*(t))+(99.67782004999999)

Generation 525, Farm 14, MDD 44, Fitness 16.213653288795477:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901722007E-08))*(t))+(&G000000009037112005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01222007))*(t))+(99.67817994000001)

Generation 527, Farm 14, MDD 46, Fitness 16.207367601822469:
(((((-t))*(ArcSinh(&G000000006765177027E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009007132005E-08)
(((((-t))*(ArcSinh(67.34127972)))+(50.01024007))*(t))+(99.92162005)

Generation 530, Farm 14, MDD 49, Fitness 16.178126640374312:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.01024007))*(t))+(99.92182004999999)

Generation 538, Farm 14, MDD 57, Fitness 16.178122581356028:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009007112805E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.01024007))*(t))+(99.92182805)

Generation 543, Farm 14, MDD 62, Fitness 16.175855861914016:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009037132005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01024007))*(t))+(99.67837994)

Generation 544, Farm 14, MDD 63, Fitness 16.17584769658928:
(((((-t))*(ArcSinh(&G000000006766177007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009037212005E-08)
(((((-t))*(ArcSinh(67.33872006999999)))+(50.01024007))*(t))+(99.67782004999999)

Generation 545, Farm 14, MDD 64, Fitness 16.174195934355915:
(((((-t))*(ArcSinh(&G000000006762137007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.37832007)))+(50.01024007))*(t))+(99.92182004999999)

Generation 552, Farm 14, MDD 71, Fitness 16.17419569068214:
(((((-t))*(ArcSinh(&G000000006762137007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009007112055E-08)
(((((-t))*(ArcSinh(67.37832007)))+(50.01024007))*(t))+(99.92182054)

Generation 557, Farm 14, MDD 76, Fitness 16.101012910455552:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005900324007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.00375992))*(t))+(99.92182004999999)

Generation 559, Farm 14, MDD 78, Fitness 16.093849524174196:
(((((-t))*(ArcSinh(&G000000006764177007E-08)))+(&G000000005901924007E-08))*(t))+(&G000000009007112805E-08)
(((((-t))*(ArcSinh(67.35872007)))+(50.01024007))*(t))+(99.92182805)

Generation 560, Farm 14, MDD 79, Fitness 16.089045450374293:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005900024007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.00024007))*(t))+(99.92182004999999)

Generation 566, Farm 14, MDD 85, Fitness 16.089042406098663:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005900024007E-08))*(t))+(&G000000009007112605E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.00024007))*(t))+(99.92182605000001)

Generation 568, Farm 14, MDD 87, Fitness 16.086037138832836:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005900024007E-08))*(t))+(&G000000009007712005E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.00024007))*(t))+(99.92782004999999)

Generation 569, Farm 14, MDD 88, Fitness 16.086034166557358:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005900024007E-08))*(t))+(&G000000009007712605E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.00024007))*(t))+(99.92782604999999)

Generation 570, Farm 14, MDD 89, Fitness 16.081015177413242:
(((((-t))*(ArcSinh(&G000000006762177007E-08)))+(&G000000005900024007E-08))*(t))+(&G000000009006112605E-08)
(((((-t))*(ArcSinh(67.37872007)))+(50.00024007))*(t))+(99.93817394)

Generation 575, Farm 14, MDD 94, Fitness 16.068677772826192:
(((((-t))*(ArcSinh(&G000000006762170800E-08)))+(&G000000005900324007E-08))*(t))+(&G000000009000112005E-08)
(((((-t))*(ArcSinh(67.37879199)))+(50.00375992))*(t))+(99.99817994)

Generation 576, Farm 14, MDD 95, Fitness 16.027792082378369:
(((((-t))*(ArcSinh(&G000000006762977007E-08)))+(&G000000005900004007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.37072007)))+(50.00004007))*(t))+(99.92182004999999)

Generation 587, Farm 14, MDD 94, Fitness 16.018350782411893:
(((((-t))*(ArcSinh(&G000000006762777007E-08)))+(&G000000005900024007E-08))*(t))+(&G000000009000112005E-08)
(((((-t))*(ArcSinh(67.37272007)))+(50.00024007))*(t))+(99.99817994)

Generation 588, Farm 14, MDD 93, Fitness 16.004769114084052:
(((((-t))*(ArcSinh(&G000000006764177007E-08)))+(&G000000005900024107E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.35872007)))+(50.00024192))*(t))+(99.92182004999999)

Generation 598, Farm 14, MDD 83, Fitness 16.004010600051256:
(((((-t))*(ArcSinh(&G000000006764077007E-08)))+(&G000000005900024107E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcSinh(67.35927992000001)))+(50.00024192))*(t))+(99.92182004999999)

Generation 599, Farm 14, MDD 82, Fitness 16.003788736634704:
(((((-t))*(ArcCosh(&G000000006762977007E-08)))+(&G000000005900004007E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcCosh(67.37072007)))+(50.00004007))*(t))+(99.92182004999999)

Generation 606, Farm 14, MDD 75, Fitness 16.003788735953606:
(((((-t))*(ArcCosh(&G000000006762977007E-08)))+(&G000000005900004006E-08))*(t))+(&G000000009007112005E-08)
(((((-t))*(ArcCosh(67.37072007)))+(50.00004006))*(t))+(99.92182004999999)

Generation 607, Farm 14, MDD 74, Fitness 16.002326325382327:
(((((-t))*(ArcSinh(&G000000006763177007E-08)))+(&G000000005900024007E-08))*(t))+(&G000000009001112705E-08)
(((((-t))*(ArcSinh(67.36127992)))+(50.00024007))*(t))+(99.98182794)

Generation 619, Farm 24, MDD 62, Fitness 15.974504203155227:
(((((-t))*(&G000000000499206007E-08))+(&G000000004900204000E-08))*(t))+(&G000000009085024107E-08)
(((((-t))*(4.90206007))+(49.99795999))*(t))+(99.14024191999999)

Generation 629, Farm 24, MDD 52, Fitness 15.970853276239254:
(((((-t))*(&G000000000499206207E-08))+(&G000000004900204000E-08))*(t))+(&G000000009085024107E-08)
(((((-t))*(4.90206207))+(49.99795999))*(t))+(99.14024191999999)

Generation 636, Farm 24, MDD 45, Fitness 15.451331356035542:
(((((-t))*(&G000000000499246007E-08))+(&G000000004900204000E-08))*(t))+(&G000000009085024107E-08)
(((((-t))*(4.90246007))+(49.99795999))*(t))+(99.14024191999999)

Generation 645, Farm 24, MDD 36, Fitness 15.436095485679317:
(((((-t))*(&G000000000499248007E-08))+(&G000000004900204000E-08))*(t))+(&G000000009085024107E-08)
(((((-t))*(4.90248007))+(49.99795999))*(t))+(99.14024191999999)

Generation 649, Farm 24, MDD 32, Fitness 15.435471006287488:
(((((-t))*(&G000000000499248097E-08))+(&G000000004900204000E-08))*(t))+(&G000000009085024107E-08)
(((((-t))*(4.90248092))+(49.99795999))*(t))+(99.14024191999999)

Generation 652, Farm 24, MDD 31, Fitness 15.371799231400445:
(((((-t))*(&G000000000499296207E-08))+(&G000000004900204000E-08))*(t))+(&G000000009085024107E-08)
(((((-t))*(4.90293792))+(49.99795999))*(t))+(99.14024191999999)

Generation 659, Farm 24, MDD 38, Fitness 15.33270216660673:
(((((-t))*(&G000000000499276007E-08))+(&G000000004900204000E-08))*(t))+(&G000000009085024107E-08)
(((((-t))*(4.90273992))+(49.99795999))*(t))+(99.14024191999999)

Generation 669, Farm 24, MDD 48, Fitness 15.262123532840372:
(((((-t))*(&G000000000499296207E-08))+(&G000000004900204000E-08))*(t))+(&G000000009075324107E-08)
(((((-t))*(4.90293792))+(49.99795999))*(t))+(99.25624191999999)

Generation 673, Farm 24, MDD 52, Fitness 15.090240175486297:
(((((-t))*(&G000000000499296207E-08))+(&G000000004900204000E-08))*(t))+(&G000000009035024107E-08)
(((((-t))*(4.90293792))+(49.99795999))*(t))+(99.65975807)

Generation 679, Farm 24, MDD 58, Fitness 15.073867473711688:
(((((-t))*(&G000000000499291207E-08))+(&G000000004900204000E-08))*(t))+(&G000000009035024107E-08)
(((((-t))*(4.90298207))+(49.99795999))*(t))+(99.65975807)

Generation 684, Farm 24, MDD 63, Fitness 15.073864339511369:
(((((-t))*(&G000000000499291208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009035024107E-08)
(((((-t))*(4.90298208))+(49.99795999))*(t))+(99.65975807)

Generation 692, Farm 24, MDD 71, Fitness 15.07372669215486:
(((((-t))*(&G000000000499291257E-08))+(&G000000004900204000E-08))*(t))+(&G000000009035024107E-08)
(((((-t))*(4.90298252))+(49.99795999))*(t))+(99.65975807)

Generation 697, Farm 24, MDD 76, Fitness 15.069226231978469:
(((((-t))*(&G000000000499290207E-08))+(&G000000004900204000E-08))*(t))+(&G000000009035024107E-08)
(((((-t))*(4.90299792))+(49.99795999))*(t))+(99.65975807)

Generation 704, Farm 24, MDD 83, Fitness 15.064304314266362:
(((((-t))*(&G000000000499391208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009035024107E-08)
(((((-t))*(4.90301791))+(49.99795999))*(t))+(99.65975807)

Generation 714, Farm 24, MDD 93, Fitness 15.055008394394413:
(((((-t))*(&G000000000499399208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009035024107E-08)
(((((-t))*(4.90309791))+(49.99795999))*(t))+(99.65975807)

Generation 722, Farm 24, MDD 99, Fitness 15.051096483333577:
(((((-t))*(&G000000000499399208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009005024107E-08)
(((((-t))*(4.90309791))+(49.99795999))*(t))+(99.94024192000001)

Generation 727, Farm 24, MDD 94, Fitness 15.05108582346015:
(((((-t))*(&G000000000499399208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009005020107E-08)
(((((-t))*(4.90309791))+(49.99795999))*(t))+(99.94020192000001)

Generation 733, Farm 24, MDD 88, Fitness 15.047560449333689:
(((((-t))*(&G000000000499399208E-08))+(&G000000004900004000E-08))*(t))+(&G000000009005024107E-08)
(((((-t))*(4.90309791))+(49.99995999))*(t))+(99.94024192000001)

Generation 736, Farm 24, MDD 85, Fitness 15.047549789460247:
(((((-t))*(&G000000000499399208E-08))+(&G000000004900004000E-08))*(t))+(&G000000009005020107E-08)
(((((-t))*(4.90309791))+(49.99995999))*(t))+(99.94020192000001)

Generation 740, Farm 24, MDD 81, Fitness 15.0377891153337:
(((((-t))*(&G000000000499399208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009025024107E-08)
(((((-t))*(4.90309791))+(49.99795999))*(t))+(99.74024192)

Generation 744, Farm 24, MDD 77, Fitness 15.037788294573044:
(((((-t))*(&G000000000499399208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009025024807E-08)
(((((-t))*(4.90309791))+(49.99795999))*(t))+(99.74024807000001)

Generation 745, Farm 24, MDD 76, Fitness 15.005313594780038:
(((((-t))*(&G000000000499379208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009005020107E-08)
(((((-t))*(4.90329791))+(49.99795999))*(t))+(99.94020192000001)

Generation 753, Farm 24, MDD 68, Fitness 15.003969423880006:
(((((-t))*(&G000000000499379208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009002020107E-08)
(((((-t))*(4.90329791))+(49.99795999))*(t))+(99.97979807)

Generation 769, Farm 24, MDD 52, Fitness 15.003966411539993:
(((((-t))*(&G000000000499379208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009002070107E-08)
(((((-t))*(4.90329791))+(49.99795999))*(t))+(99.97920191999999)

Generation 773, Farm 24, MDD 48, Fitness 15.003966394395974:
(((((-t))*(&G000000000499379208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009002080107E-08)
(((((-t))*(4.90329791))+(49.99795999))*(t))+(99.97919807)

Generation 783, Farm 24, MDD 38, Fitness 15.003966161444:
(((((-t))*(&G000000000499379208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009002580107E-08)
(((((-t))*(4.90329791))+(49.99795999))*(t))+(99.97480192)

Generation 786, Farm 24, MDD 35, Fitness 15.003962841815611:
(((((-t))*(&G000000000499379208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009002420807E-08)
(((((-t))*(4.90329791))+(49.99795999))*(t))+(99.97579192000001)

Generation 787, Farm 24, MDD 34, Fitness 15.003897374055967:
(((((-t))*(&G000000000499379208E-08))+(&G000000004900202000E-08))*(t))+(&G000000009002080107E-08)
(((((-t))*(4.90329791))+(49.99797999))*(t))+(99.97919807)

Generation 800, Farm 24, MDD 39, Fitness 15.003842686645045:
(((((-t))*(&G000000000499369208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009002080207E-08)
(((((-t))*(4.90330208))+(49.99795999))*(t))+(99.97919792)

Generation 812, Farm 24, MDD 51, Fitness 15.003842686618665:
(((((-t))*(&G000000000499369208E-08))+(&G000000004900204000E-08))*(t))+(&G000000009002080206E-08)
(((((-t))*(4.90330208))+(49.99795999))*(t))+(99.97919793)

Generation 824, Farm 24, MDD 63, Fitness 15.003570651964912:
(((((-t))*(&G000000000499369208E-08))+(&G000000004900104000E-08))*(t))+(&G000000009002080207E-08)
(((((-t))*(4.90330208))+(49.99804))*(t))+(99.97919792)

Generation 834, Farm 24, MDD 73, Fitness 15.003570651885722:
(((((-t))*(&G000000000499369208E-08))+(&G000000004900104000E-08))*(t))+(&G000000009002080204E-08)
(((((-t))*(4.90330208))+(49.99804))*(t))+(99.97919795)

Generation 835, Farm 24, MDD 74, Fitness 15.000306652645014:
(((((-t))*(&G000000000499369208E-08))+(&G000000004900004000E-08))*(t))+(&G000000009002080207E-08)
(((((-t))*(4.90330208))+(49.99995999))*(t))+(99.97919792)

Generation 838, Farm 24, MDD 77, Fitness 15.000306652460209:
(((((-t))*(&G000000000499369208E-08))+(&G000000004900004000E-08))*(t))+(&G000000009002080200E-08)
(((((-t))*(4.90330208))+(49.99995999))*(t))+(99.97919799)

Generation 839, Farm 24, MDD 78, Fitness 15.000287427315021:
(((((-t))*(&G000000000499369298E-08))+(&G000000004900004000E-08))*(t))+(&G000000009002080236E-08)
(((((-t))*(4.90330291))+(49.99995999))*(t))+(99.97919766)

```
Generation 844, Farm 24, MDD 83, Fitness 15.000196546968338:
(((((-t))*(&G0000000000499369908E-08))+(&G000000004900004000E-08))*(t))+(&G000000009002080206E-08)
(((((-t))*(4.90330991))+(49.99995999))*(t))+(99.97919793)

Generation 850, Farm 24, MDD 89, Fitness 15.0001965464898:
(((((-t))*(&G0000000000499369908E-08))+(&G000000004900004000E-08))*(t))+(&G000000009002080203E-08)
(((((-t))*(4.90330991))+(49.99995999))*(t))+(99.97919795999999)

Generation 853, Farm 24, MDD 92, Fitness 15.000195186288336:
(((((-t))*(&G0000000000499369908E-08))+(&G000000004900000000E-08))*(t))+(&G000000009002080206E-08)
(((((-t))*(4.90330991))+(49.99999999))*(t))+(99.97919793)

Generation 858, Farm 24, MDD 97, Fitness 15.000184426056336:
(((((-t))*(&G0000000000499369908E-08))+(&G000000004900004000E-08))*(t))+(&G000000009002000206E-08)
(((((-t))*(4.90330991))+(49.99995999))*(t))+(99.97999793)

Generation 863, Farm 212, MDD 98, Fitness 13.069604410926377:
(&G000000009010000042E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499290562E-08)*(t))))
(99.80000042)+((t)*((49.99996096)-((4.90299462)*(t))))

Generation 870, Farm 212, MDD 91, Fitness 13.069583644736468:
(&G000000009010000042E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499290566E-08)*(t))))
(99.80000042)+((t)*((49.99996096)-((4.90299466)*(t))))

Generation 873, Farm 212, MDD 88, Fitness 13.063507233367453:
(&G000000009025003040E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499290562E-08)*(t))))
(99.74003958999999)+((t)*((49.99996096)-((4.90299462)*(t))))

Generation 878, Farm 212, MDD 83, Fitness 13.061299976776811:
(&G000000009025003042E-08)+((t)*((&G000000004900003040E-08)-((&G0000000000499390000E-08)*(t))))
(99.74003956999999)+((t)*((49.9999604)-((4.903)*(t))))

Generation 880, Farm 212, MDD 81, Fitness 13.017954257833251:
(&G000000009010000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499370500E-08)*(t))))
(99.80000357)+((t)*((49.99996096)-((4.90320599)*(t))))

Generation 894, Farm 212, MDD 67, Fitness 13.017853105575663:
(&G000000009010000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499370100E-08)*(t))))
(99.80000357)+((t)*((49.99996096)-((4.90320199)*(t))))

Generation 897, Farm 212, MDD 64, Fitness 13.011431158183381:
(&G000000009013000342E-08)+((t)*((&G000000004900003053E-08)-((&G0000000000499371500E-08)*(t))))
(99.83999642000001)+((t)*((49.99996056)-((4.903214)*(t))))

Generation 910, Farm 212, MDD 51, Fitness 13.011430786495364:
(&G000000009013000342E-08)+((t)*((&G000000004900003653E-08)-((&G0000000000499371500E-08)*(t))))
(99.83999642000001)+((t)*((49.99996656)-((4.903214)*(t))))

Generation 913, Farm 212, MDD 48, Fitness 13.008186655266682:
(&G000000009019000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499370500E-08)*(t))))
(99.89999641999999)+((t)*((49.99996096)-((4.90320599)*(t))))

Generation 924, Farm 212, MDD 37, Fitness 13.004713859460171:
(&G000000009019000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499377500E-08)*(t))))
(99.89999641999999)+((t)*((49.99996096)-((4.903274)*(t))))

Generation 939, Farm 212, MDD 38, Fitness 13.004486542940175:
(&G000000009019000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499375500E-08)*(t))))
(99.89999641999999)+((t)*((49.99996096)-((4.903254)*(t))))

Generation 955, Farm 212, MDD 54, Fitness 13.004486527078811:
(&G000000009019000362E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499375500E-08)*(t))))
(99.89999662)+((t)*((49.99996096)-((4.903254)*(t))))

Generation 957, Farm 212, MDD 56, Fitness 13.004486413604175:
(&G000000009019000342E-08)+((t)*((&G000000004900003293E-08)-((&G0000000000499375500E-08)*(t))))
(99.89999641999999)+((t)*((49.99996296)-((4.903254)*(t))))

Generation 961, Farm 212, MDD 60, Fitness 13.004485524457598:
(&G000000009019000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499375507E-08)*(t))))
(99.89999641999999)+((t)*((49.99996096)-((4.90325407)*(t))))

Generation 967, Farm 212, MDD 66, Fitness 13.004485235803182:
(&G000000009019000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499375509E-08)*(t))))
(99.89999641999999)+((t)*((49.99996096)-((4.90325409)*(t))))

Generation 968, Farm 212, MDD 67, Fitness 13.004462557236739:
(&G000000009019000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499375400E-08)*(t))))
(99.89999641999999)+((t)*((49.99996096)-((4.90325599)*(t))))

Generation 974, Farm 212, MDD 73, Fitness 13.004445500674342:
(&G000000009019000342E-08)+((t)*((&G000000004900003093E-08)-((&G0000000000499375000E-08)*(t))))
(99.89999641999999)+((t)*((49.99996096)-((4.90325999)*(t))))
```

```
Generation 975, Farm 212, MDD 74, Fitness 13.004445443230832:
(&G000000009019000340E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499375007E-08)*(t))))
(99.89999640000001)+((t)*((49.99996096)-((4.90325992)*(t))))

Generation 977, Farm 212, MDD 76, Fitness 13.004445371338342:
(&G000000009019000342E-08)+((t)*((&G000000004900003293E-08)-((&G000000000499375000E-08)*(t))))
(99.89999641999999)+((t)*((49.99996296)-((4.90325999)*(t))))

Generation 979, Farm 212, MDD 78, Fitness 13.003300399740187:
(&G000000009008000342E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499375500E-08)*(t))))
(99.91999642)+((t)*((49.99996096)-((4.903254)*(t))))

Generation 982, Farm 212, MDD 81, Fitness 13.002914261545135:
(&G000000009005000342E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499375500E-08)*(t))))
(99.94000357)+((t)*((49.99996096)-((4.903254)*(t))))

Generation 990, Farm 212, MDD 89, Fitness 13.002754106245106:
(&G000000009000700342E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499377500E-08)*(t))))
(99.99200356999999)+((t)*((49.99996096)-((4.903274)*(t))))

Generation 1000, Farm 212, MDD 99, Fitness 13.001597284945104:
(&G000000009005790342E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499377500E-08)*(t))))
(99.94700357000001)+((t)*((49.99996096)-((4.903274)*(t))))

Generation 1004, Farm 212, MDD 97, Fitness 13.001597283980473:
(&G000000009005790332E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499377500E-08)*(t))))
(99.94700362)+((t)*((49.99996096)-((4.903274)*(t))))

Generation 1006, Farm 212, MDD 95, Fitness 13.001579115780187:
(&G000000009005700342E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499377500E-08)*(t))))
(99.94799642)+((t)*((49.99996096)-((4.903274)*(t))))

Generation 1008, Farm 212, MDD 93, Fitness 13.001389126465048:
(&G000000009005790332E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499377000E-08)*(t))))
(99.94700362)+((t)*((49.99996096)-((4.90327999)*(t))))

Generation 1012, Farm 212, MDD 89, Fitness 13.001389120569931:
(&G000000009005790312E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499377000E-08)*(t))))
(99.94700382000001)+((t)*((49.99996096)-((4.90327999)*(t))))

Generation 1014, Farm 212, MDD 87, Fitness 13.001362856547638:
(&G000000009005707342E-08)+((t)*((&G000000004900003093E-08)-((&G000000000499377000E-08)*(t))))
(99.94792357)+((t)*((49.99996096)-((4.90327999)*(t))))

Generation 1023, Farm 212, MDD 78, Fitness 13.00093093462293:
(&G000000009000790332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499379500E-08)*(t))))
(99.99299637)+((t)*((49.99996003)-((4.903294)*(t))))

Generation 1030, Farm 212, MDD 71, Fitness 13.000799342624626:
(&G000000009000790332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499379400E-08)*(t))))
(99.99299637)+((t)*((49.99996003)-((4.90329599)*(t))))

Generation 1048, Farm 212, MDD 53, Fitness 13.000797482818415:
(&G000000009000795332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499379400E-08)*(t))))
(99.99294362000001)+((t)*((49.99996003)-((4.90329599)*(t))))

Generation 1049, Farm 212, MDD 52, Fitness 13.000796076248802:
(&G000000009000799332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499379400E-08)*(t))))
(99.99290362000001)+((t)*((49.99996003)-((4.90329599)*(t))))

Generation 1053, Farm 212, MDD 48, Fitness 13.000677465993414:
(&G000000009000790332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499379200E-08)*(t))))
(99.99299637)+((t)*((49.99996003)-((4.90329799)*(t))))

Generation 1059, Farm 212, MDD 42, Fitness 13.00062365323282:
(&G000000009001790332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499379400E-08)*(t))))
(99.98700362)+((t)*((49.99996003)-((4.90329599)*(t))))

Generation 1066, Farm 212, MDD 35, Fitness 13.000373824732915:
(&G000000009000790332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499369400E-08)*(t))))
(99.99299637)+((t)*((49.99996003)-((4.903304)*(t))))

Generation 1075, Farm 212, MDD 34, Fitness 13.000373737977965:
(&G000000009000790732E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499369400E-08)*(t))))
(99.99299237)+((t)*((49.99996003)-((4.903304)*(t))))

Generation 1080, Farm 212, MDD 39, Fitness 13.000372526768516:
(&G000000009000796332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499369400E-08)*(t))))
(99.99293637)+((t)*((49.99996003)-((4.903304)*(t))))

Generation 1086, Farm 212, MDD 45, Fitness 13.00037182133384:
(&G000000009000799332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499369400E-08)*(t))))
(99.99290362000001)+((t)*((49.99996003)-((4.903304)*(t))))
```

Generation 1089, Farm 212, MDD 48, Fitness 13.000357030185969:
(&G000000009000710732E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499369400E-08)*(t))))
(99.99219237)+((t)*((49.99996003)-((4.903304)*(t))))

Generation 1094, Farm 212, MDD 53, Fitness 13.000334567577848:
(&G000000009000890332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499369400E-08)*(t))))
(99.99100362)+((t)*((49.99996003)-((4.903304)*(t))))

Generation 1096, Farm 212, MDD 55, Fitness 13.000039802952942:
(&G000000009000790332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499367400E-08)*(t))))
(99.99299637)+((t)*((49.99996003)-((4.903324)*(t))))

Generation 1114, Farm 212, MDD 73, Fitness 13.000038786353537:
(&G000000009000790332E-08)+((t)*((&G000000004900002003E-08)-((&G000000000499367400E-08)*(t))))
(99.99299637)+((t)*((49.99997996)-((4.903324)*(t))))

Generation 1115, Farm 212, MDD 74, Fitness 13.000038444993537:
(&G000000009000790332E-08)+((t)*((&G000000004900000003E-08)-((&G000000000499367400E-08)*(t))))
(99.99299637)+((t)*((49.99999996)-((4.903324)*(t))))

Generation 1117, Farm 212, MDD 76, Fitness 13.00000195721787:
(&G000000009000090332E-08)+((t)*((&G000000004900003003E-08)-((&G000000000499367400E-08)*(t))))
(99.99900362)+((t)*((49.99996003)-((4.903324)*(t))))

Generation 1120, Farm 212, MDD 79, Fitness 13.000001703306465:
(&G000000009000090332E-08)+((t)*((&G000000004900003303E-08)-((&G000000000499367400E-08)*(t))))
(99.99900362)+((t)*((49.99996396)-((4.903324)*(t))))

Generation 1124, Farm 212, MDD 83, Fitness 13.000001580123868:
(&G000000009000090332E-08)+((t)*((&G000000004900003603E-08)-((&G000000000499367400E-08)*(t))))
(99.99900362)+((t)*((49.99996603)-((4.903324)*(t))))

Generation 1131, Farm 212, MDD 90, Fitness 13.000000630266465:
(&G000000009000090332E-08)+((t)*((&G000000004900000603E-08)-((&G000000000499367400E-08)*(t))))
(99.99900362)+((t)*((49.99999396)-((4.903324)*(t))))

Generation 1132, Farm 212, MDD 91, Fitness 13.000000612653869:
(&G000000009000090332E-08)+((t)*((&G000000004900000303E-08)-((&G000000000499367400E-08)*(t))))
(99.99900362)+((t)*((49.99999603)-((4.903324)*(t))))

Generation 1134, Farm 212, MDD 93, Fitness 13.000000599258465:
(&G000000009000090332E-08)+((t)*((&G000000004900000003E-08)-((&G000000000499367400E-08)*(t))))
(99.99900362)+((t)*((49.99999996)-((4.903324)*(t))))

Generation 1141, Farm 212, MDD 100, Fitness 13.000000565066433:
(&G000000009000090332E-08)+((t)*((&G000000004900000003E-08)-((&G000000000499367404E-08)*(t))))
(99.99900362)+((t)*((49.99999996)-((4.90332404)*(t))))

Generation 1144, Farm 212, MDD 97, Fitness 13.000000549530812:
(&G000000009000090332E-08)+((t)*((&G000000004900000003E-08)-((&G000000000499367406E-08)*(t))))
(99.99900362)+((t)*((49.99999996)-((4.90332406)*(t))))

Generation 1151, Farm 212, MDD 90, Fitness 13.000000549392769:
(&G000000009000090302E-08)+((t)*((&G000000004900000003E-08)-((&G000000000499367406E-08)*(t))))
(99.99900397)+((t)*((49.99999996)-((4.90332406)*(t))))

Generation 1157, Farm 212, MDD 84, Fitness 13.00000047569643:
(&G000000009000090332E-08)+((t)*((&G000000004900000604E-08)-((&G000000000499367440E-08)*(t))))
(99.99900362)+((t)*((49.99999395)-((4.9033244)*(t))))

Generation 1167, Farm 212, MDD 74, Fitness 13.000000443046805:
(&G000000009000090302E-08)+((t)*((&G000000004900000003E-08)-((&G000000000499367430E-08)*(t))))
(99.99900397)+((t)*((49.99999996)-((4.90332439)*(t))))

Generation 1172, Farm 212, MDD 69, Fitness 13.000000394075501:
(&G000000009000099332E-08)+((t)*((&G000000004900000604E-08)-((&G000000000499367440E-08)*(t))))
(99.99909637)+((t)*((49.99999395)-((4.9033244)*(t))))

Generation 1175, Farm 212, MDD 66, Fitness 13.00000024659243:
(&G000000009000050332E-08)+((t)*((&G000000004900000604E-08)-((&G000000000499367440E-08)*(t))))
(99.99940362)+((t)*((49.99999395)-((4.9033244)*(t))))

Generation 1189, Farm 212, MDD 52, Fitness 13.000000189689843:
(&G000000009000050332E-08)+((t)*((&G000000004900000604E-08)-((&G000000000499367450E-08)*(t))))
(99.99940362)+((t)*((49.99999395)-((4.90332459)*(t))))

Generation 1194, Farm 212, MDD 47, Fitness 13.000000172519842:
(&G000000009000050332E-08)+((t)*((&G000000004900000404E-08)-((&G000000000499367450E-08)*(t))))
(99.99940362)+((t)*((49.99999595)-((4.90332459)*(t))))

Generation 1199, Farm 212, MDD 42, Fitness 13.0000001724901:
(&G000000009000050338E-08)+((t)*((&G000000004900000404E-08)-((&G000000000499367450E-08)*(t))))
(99.99940368)+((t)*((49.99999595)-((4.90332459)*(t))))

```
Generation 1212, Farm 212, MDD 31, Fitness 13.000000166389684:
(&G000000009000051338E-08)+((t)*((&G000000004900000404E-08)-((&G000000000499367450E-08)*(t))))
(99.99941631)+((t)*((49.99999595)-((4.90332459)*(t))))

Generation 1214, Farm 212, MDD 33, Fitness 13.000000149151065:
(&G000000009000055332E-08)+((t)*((&G000000004900000404E-08)-((&G000000000499367450E-08)*(t))))
(99.99945637)+((t)*((49.99999595)-((4.90332459)*(t))))

Generation 1220, Farm 212, MDD 39, Fitness 13.0000001392389:
(&G000000009000058338E-08)+((t)*((&G000000004900000404E-08)-((&G000000000499367450E-08)*(t))))
(99.99948368)+((t)*((49.99999595)-((4.90332459)*(t))))

Generation 1227, Farm 212, MDD 46, Fitness 13.000000130537842:
(&G000000009000030332E-08)+((t)*((&G000000004900000604E-08)-((&G000000000499367450E-08)*(t))))
(99.99960362)+((t)*((49.99999395)-((4.90332459)*(t))))

Generation 1235, Farm 212, MDD 54, Fitness 13.000000130332483:
(&G000000009000030332E-08)+((t)*((&G000000004900000602E-08)-((&G000000000499367450E-08)*(t))))
(99.99960362)+((t)*((49.99999397)-((4.90332459)*(t))))

Generation 1246, Farm 146, MDD 65, Fitness 13.000000099506172:
(&G000000009000000870E-08)-((((&G000000000499367489E-08)*(t))-(&G000000004900001901E-08))*(t))
(99.9999912)-((((4.90332489)*(t))-(49.99998998))*(t))

Generation 1255, Farm 212, MDD 74, Fitness 13.000000099427842:
(&G000000009000030332E-08)+((t)*((&G000000004900000004E-08)-((&G000000000499367450E-08)*(t))))
(99.99960362)+((t)*((49.99999995)-((4.90332459)*(t))))

Generation 1256, Farm 146, MDD 75, Fitness 13.000000010482173:
(&G000000009000008870E-08)-((((&G000000000499367489E-08)*(t))-(&G000000004900000201E-08))*(t))
(99.9999112)-((((4.90332489)*(t))-(49.99999798))*(t))

Generation 1260, Farm 146, MDD 79, Fitness 13.000000010447918:
(&G000000009000008870E-08)-((((&G000000000499367489E-08)*(t))-(&G000000005900000201E-08))*(t))
(99.9999112)-((((4.90332489)*(t))-(50.00000201))*(t))

Generation 1267, Farm 215, MDD 86, Fitness 9:
(((((-t))*(&G000000000499367590E-08))*(t))+(&G000000019000000000E-08))+((t)*(&G000000005900000000E-08))
(((((-t))*(4.903325))*(t))+(100))+((t)*(50))

Generation 1320, Farm 24, MDD 61, Fitness 7.5:
(((((-t))*(&G000000000499367590E-08))+(&G000000005900000000E-08))*(t))+(&G000000019000000000E-08)
(((((-t))*(4.903325))+(50))*(t))+(100)

Generation 1457, Farm 212, MDD 64, Fitness 6.5:
(&G000000019000000000E-08)+((t)*((&G000000005900000000E-08)-((&G000000000499367590E-08)*(t))))
(100)+((t)*((50)-((4.903325)*(t))))
```

13.4.9 Round 3, Simulation 2

```
Generation 1, Farm 46, MDD 120, Fitness 5.0331774420245896:
Sin((&G000000000000008513E-08)+(phi)):
(Tan((-(phi)*((&G000000000000002154E-08)/((&G080813602249931892E-08)/(&G000000000000000001E-08))))))-(&G000000000000002869E-08)
Sin((8.583E-05)+(phi)):(Tan((-(phi)*((2.144E-05)/((808166022.490389)/(1E-08))))))-(2.869E-05)

Generation 4, Farm 157, MDD 123, Fitness 4.7104317927478299:
(phi)*(Tan(((&G000000000035604726E-08)/(&G000071850226028031E-08))/((Sinh(&G000000000006064138E-08))*(&G000000000000047965E-08)))):
((-phi))*(&G000000000000051051E-08)
(phi)*(Tan(((0.34604773)/(788597.73971961))/((Sinh(0.06064168))*(0.00047065)))):((-phi))*(0.00058058)

Generation 5, Farm 46, MDD 124, Fitness 4.1690066623665496:
Sin((&G000900000000008513E-08)+(phi)):
(Tan((-(phi)*((&G000000000000002150E-08)/((&G080813602249931892E-08)/(&G000000000000000001E-08))))))-(&G000000000000002869E-08)
Sin((9999999.99991416)+(phi)):(Tan((-(phi)*((2.14E-05)/((808166022.490389)/(1E-08))))))-(2.869E-05)

Generation 7, Farm 46, MDD 126, Fitness 4.168999238059345:
Sin((&G000900000000008913E-08)+(phi)):
(Tan((-(phi)*((&G000000000000002150E-08)/((&G080813602249931892E-08)/(&G000000000000000001E-08))))))-(&G000000000000002869E-08)
Sin((9999999.999910161)+(phi)):(Tan((-(phi)*((2.14E-05)/((808166022.490389)/(1E-08))))))-(2.869E-05)

Generation 8, Farm 147, MDD 127, Fitness 2.9901683008504598:
Cos(phi):
Sinh((((-ArcTan((&G000000036209608426E-08)*((&G000000000000013838E-08)^(Cos(phi))))))/(&G000000135507329529E-08))*(&G000000000000490318E-08))
Cos(phi):Sinh((((-ArcTan((337.90608426)*((0.00016831)^(Cos(phi))))))/(1654.07629429))*(0.00499611))

Generation 10, Farm 134, MDD 129, Fitness 2.7809561291009706:
(Tanh(Cos(phi)))/(&G000000000039094245E-08):
((&G000000000008821741E-08)^(ArcTan(Cosh(phi))))/(&G064754951965362250E-08)
(Tanh(Cos(phi)))/(0.30095754):((0.08821241)^(ArcTan(Cosh(phi))))/(647449410.6566226)
```

```
Generation 11, Farm 134, MDD 130, Fitness 2.7809091196427338:
(Tanh(Cos(phi)))/(&G000000000039094245E-08):
(((&G000000000004730066E-08)+(&G0003312656681449 98E-08))/(ArcTan(Cosh(phi))))/(&G0647549519653622 50E-08)
(Tanh(Cos(phi)))/(0.30095754):((((0.04760066)+(3617346.68155091))/(ArcTan(Cosh(phi))))/(647449410.6566226)

Generation 15, Farm 134, MDD 134, Fitness 2.7809091159976145:
(Tanh(Cos(phi)))/(&G000000000039094245E-08):
(((&G000000003004730066E-08)+(&G0003312656686449 98E-08))/(ArcTan(Cosh(phi))))/(&G0647549519653622 50E-08)
(Tanh(Cos(phi)))/(0.30095754):(((39.95239933)+(3617346.68644908))/(ArcTan(Cosh(phi))))/(647449410.6566226)

Generation 16, Farm 134, MDD 135, Fitness 2.7807974267016706:
(Tanh(Cos(phi)))/(&G000000000039090245E-08):
((ArcCosh(&G000027417689795100E-08))^(ArcTan(Cosh(phi))))/(&G06475490196536225 0E-08)
(Tanh(Cos(phi)))/(0.30099754):((ArcCosh(275873.107058))^(ArcTan(Cosh(phi))))/(647449989.3433774)

Generation 17, Farm 10, MDD 136, Fitness 2.2506618312014681:
(Tan(Tan((&G000000000000000003E-08)*((&G000000000000025156E-08)-(&G000009234248879409E-08))))+((Cos((-phi)))+(&G000000000000000995E-08)):
Sin(phi)
(Tan(Tan((3E-08)*((0.00025853)-(97642.48870409001))))+((Cos((-phi)))+(9.05E-06)):Sin(phi)

Generation 19, Farm 10, MDD 138, Fitness 2.250648490687829:
(Tan(Tan((&G000000000000000003E-08)*((&G000000000000025156E-08)-(&G000009234248879409E-08))))+((Cos((-phi)))+(&G000000000000000095E-08)):
Sin(phi)
(Tan(Tan((3E-08)*((0.00025853)-(97642.48870409001))))+((Cos((-phi)))+(9.4E-07)):Sin(phi)

Generation 21, Farm 10, MDD 140, Fitness 2.0968439015809213:
(Tan(Tan((&G000000000000000803E-08)*((&G000000000000025156E-08)-(&G000009234248879409E-08))))+((Cos((-phi)))+(&G000000000000000995E-08))
:Sin(phi)
(Tan(Tan((8.029999999999999E-06)*((0.00025853)-(97642.48870409001))))+((Cos((-phi)))+(9.05E-06)):Sin(phi)

Generation 22, Farm 10, MDD 141, Fitness 1.580130503475514:
(Tan(Tan((&G000000000000000803E-08)*((&G000000000000025156E-08)-(&G000009234248879409E-08))))+((Cos((-phi)))+(&G000000000080000995E-08)):
Sin(phi)
(Tan(Tan((8.029999999999999E-06)*((0.00025853)-(97642.48870409001))))+((Cos((-phi)))+(0.80000905)):Sin(phi)

Generation 26, Farm 10, MDD 145, Fitness 1.5758719125825742:
(Tan(Tan((&G000000000000000003E-08)*((&G000007000000025156E-08)-(&G400009234248879409E-08))))+((Cos((-phi)))+(&G000000000000000095E-08)):
Sin(phi)
(Tan(Tan((3E-08)*((79999.99974145999)-(4000097642.488704))))+((Cos((-phi)))+(9.4E-07)):Sin(phi)

Generation 30, Farm 10, MDD 149, Fitness 1.5758719125825467:
(Tan(Tan((&G000000000000000003E-08)*((&G000007000000025156E-08)-(&G400009234248879009E-08))))+((Cos((-phi)))+(&G000000000000000095E-08)):Sin(phi)
(Tan(Tan((3E-08)*((79999.99974145999)-(4000097642.4887))))+((Cos((-phi)))+(9.4E-07)):Sin(phi)

Generation 31, Farm 10, MDD 150, Fitness 1.3055715756236168:
(Tan(Tan((&G000000000000000003E-08)*((&G020007000000025156E-08)-(&G000009234248879409E-08))))+((Cos((-phi)))+((Cos((-phi)))+(&G000000000000000995E-08))):
Sin(phi)
(Tan(Tan((3E-08)*((200079999.9997415)-(97642.48470409001))))+((Cos((-phi)))+((Cos((-phi)))+(9.05E-06))):Sin(phi)

Generation 33, Farm 10, MDD 152, Fitness 0.8035333090957314:
(Tan((Cos((-phi)))+(Sin(&G000651710309610216E-08))))+((Cos((-phi)))+(&G000000000000000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.90619786))))+((Cos((-phi)))+(9.05E-06)):Sin(phi)

Generation 36, Farm 10, MDD 155, Fitness 0.80303960901545057:
(Tan((Cos((-phi)))+(Sin(&G000651710309600216E-08))))+((Cos((-phi)))+(&G000000000000000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.90600213))))+((Cos((-phi)))+(9.05E-06)):Sin(phi)

Generation 43, Farm 10, MDD 158, Fitness 0.80226304146157934:
(Tan(((Cos((-phi)))+(&G000000000000040995E-08))+(Sin(&G000651710309600516E-08))))+((Cos((-phi)))+(&G000000000000000995E-08)):Sin(phi)
(Tan(((Cos((-phi)))+(0.00040905))+(Sin(6587803.90600586))))+((Cos((-phi)))+(9.05E-06)):Sin(phi)

Generation 44, Farm 10, MDD 157, Fitness 0.7627636446439795:
(Tan((Cos((-phi)))+(Sin(&G000651710009600216E-08))))+((Cos((-phi)))+(&G000000000000000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587800.09399786))))+((Cos((-phi)))+(9.05E-06)):Sin(phi)

Generation 52, Farm 10, MDD 149, Fitness 0.51395574494887897:
(Tan((Cos((-phi)))+(Sin(&G000651710000600216E-08))))+((Cos(&G000000000080000995E-08))+(&G000000000000000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587800.00600213))))+((Cos(0.80000905))+(9.05E-06)):Sin(phi)

Generation 53, Farm 10, MDD 148, Fitness 0.3327432387648937:
(Tan((Cos((-phi)))+(Sin(&G000651710306600216E-08))))+((&G000000000000000003E-08)+(&G000000000006000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.93399786))))+((3E-08)+(0.06000905)):Sin(phi)

Generation 60, Farm 10, MDD 141, Fitness 0.33274222094742539:
(Tan((Cos((-phi)))+(Sin(&G000651710306600216E-08))))+((&G000000000000000695E-08)+(&G000000000006000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.93399786))))+((6.94E-06)+(0.06000905)):Sin(phi)

Generation 61, Farm 10, MDD 140, Fitness 0.32801291356443552:
(Tan((Cos((-phi)))+(Sin(&G000651710306600216E-08))))+((&G000000000009000003E-08)+(&G000000000006000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.93399786))))+((0.09999996)+(0.06000905)):Sin(phi)

Generation 64, Farm 10, MDD 137, Fitness 0.32277523914177048:
(Tan((Cos((-phi)))+(Sin(&G000651710305600216E-08))))+((Cos(&G000000009080000995E-08))+(&G000000000000000995E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.94600213))))+((Cos(99.19999094000001))+(9.05E-06)):Sin(phi)
```

```
Generation 69, Farm 10, MDD 132, Fitness 0.32277502583528778:
(Tan((Cos((-phi)))+(Sin(&G0006517103056002l6E-08))))+((Cos(&G00000000090800009950E-08))+(&G00000000000000006950E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.94600213))))+((Cos(99.19999094000001))+(6.94E-06)):Sin(phi)

Generation 71, Farm 10, MDD 130, Fitness 0.32277112709875688:
(Tan((Cos((-phi)))+(Sin(&G0006517103056002l6E-08))))+((Cos(&G00000000908000599 5E-08))+(&G00000000000000009950E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.94600213))))+((Cos(99.19994905))+(9.05E-06)):Sin(phi)

Generation 75, Farm 10, MDD 126, Fitness 0.32028623660218897:
(Tan((Cos((-phi)))+(Sin(&G0006517103056002l6E-08))))+((Cos(&G00000000908600099 5E-08))+(&G00000000000000009950E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.94600213))))+((Cos(99.13999094))+(9.05E-06)):Sin(phi)

Generation 83, Farm 10, MDD 118, Fitness 0.32028623510753612:
(Tan((Cos((-phi)))+(Sin(&G0006517103056002l3E-08))))+((Cos(&G00000000908600099 5E-08))+(&G00000000000000009950E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.94600216))))+((Cos(99.13999094))+(9.05E-06)):Sin(phi)

Generation 84, Farm 10, MDD 117, Fitness 0.32027804519556696:
(Tan((Cos((-phi)))+(Sin(&G0006517103057002l6E-08))))+((Cos(&G00000000908600099 5E-08))+(&G00000000000005009950E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.94799786))))+((Cos(99.13999094))+(0.00599094)):Sin(phi)

Generation 88, Farm 10, MDD 113, Fitness 0.32024494648495727:
(Tan((Cos((-phi)))+(Sin(&G0006517103056002l6E-08))))+((Cos(&G00000000978600099 5E-08))+(&G00000000000000009950E-08)):Sin(phi)
(Tan((Cos((-phi)))+(Sin(6587803.94600213))))+((Cos(92.86000905))+(9.05E-06)):Sin(phi)

Generation 90, Farm 10, MDD 111, Fitness 0.32008623660218893:
(Tan((Cos(phi))+(Sin(&G0006517103056002l6E-08))))+((Cos(&G00000000908600099 5E-08))+(&G00000000000000009950E-08)):Sin(phi)
(Tan((Cos(phi))+(Sin(6587803.94600213))))+((Cos(99.13999094))+(9.05E-06)):Sin(phi)

Generation 96, Farm 106, MDD 105, Fitness 0.30679403406574901:
Sinh(Tan(((&G0000110000000003144E-08)*(Cos(ArcTanh(&G0000000000000800008E-08))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))
Sinh(Tan(((180000.00003844)*(Cos(ArcTanh(0.00800008))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))

Generation 102, Farm 106, MDD 99, Fitness 0.30674846064618455:
Sinh(Tan(((&G0000120000000003144E-08)*(Cos(ArcTanh(&G0000000000000800008E-08))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))
Sinh(Tan(((179999.99996155)*(Cos(ArcTanh(0.00800008))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))

Generation 103, Farm 106, MDD 98, Fitness 0.30604255938373059:
Sinh(Tan(((&G0000110000000003144E-08)*(Cos(ArcTanh(&G0000000000000800108E-08))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))
Sinh(Tan(((180000.00003844)*(Cos(ArcTanh(0.008001909999999999))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))

Generation 112, Farm 106, MDD 89, Fitness 0.30604254712608575:
Sinh(Tan(((&G0000110000000003174E-08)*(Cos(ArcTanh(&G0000000000000800108E-08))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))
Sinh(Tan(((180000.00003875)*(Cos(ArcTanh(0.008001909999999999))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))

Generation 115, Farm 106, MDD 86, Fitness 0.29976684538528103:
Sinh(Tan(((&G0000110000000003144E-08)*(Cos(ArcTanh(&G0000000000000800108E-08))))+(Cos(phi)))):
(-Tanh((-(phi)*((-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))))))
Sinh(Tan(((180000.00003844)*(Cos(ArcTanh(0.008001909999999999))))+(Cos(phi)))):(-Tanh((-(phi)*((-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))))))))

Generation 117, Farm 106, MDD 84, Fitness 0.29922015369459365:
Sinh(Tan(((&G0000110000000003144E-08)*(Cos(ArcTanh(&G0000000000000800008E-08))))+(Cos(phi)))):
(-Tanh((-(phi)*(ArcTanh((-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))))))))
Sinh(Tan(((180000.00003844)*(Cos(ArcTanh(0.00800008))))+(Cos(phi)))):(-Tanh((-(phi)*(ArcTanh((-Tanh((-(phi)*(ArcTanh((-ArcSinh((-Sin(phi))))))))))))))

Generation 125, Farm 47, MDD 76, Fitness 0.22937913986712213:
((Cos(phi))-(Tanh(Log(Cosh(Cos(phi))))))*(ArcCosh(Cosh(ArcCos(Cos(phi))))):(Sin((phi)*(Sin(&G0000000000706234585E-08))))*(ArcCos(Cos(phi)))
((Cos(phi))-(Tanh(Log(Cosh(Cos(phi))))))*(ArcCosh(Cosh(ArcCos(Cos(phi))))):(Sin((phi)*(Sin(7.93764514))))*(ArcCos(Cos(phi)))

Generation 128, Farm 47, MDD 73, Fitness 0.22917913986740274:
((Cos(phi))-(Tanh(Log(Cosh(Cos(phi))))))*(ArcCos(Cos(phi))):(Sin((phi)*(Sin(&G0000000000706234585E-08))))*(ArcCos(Cos(phi)))
((Cos(phi))-(Tanh(Log(Cosh(Cos(phi))))))*(ArcCos(Cos(phi))):(Sin((phi)*(Sin(7.93764514))))*(ArcCos(Cos(phi)))

Generation 134, Farm 36, MDD 67, Fitness 0.14853569852327861:
(Exp((-Cos(phi))))*(((Exp((-Cos(phi))))*(Tan(&G000017750147449794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Cos(phi))
(Exp((-Cos(phi))))*(((Exp((-Cos(phi))))*(Tan(127401.52449295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Cos(phi))

Generation 146, Farm 36, MDD 55, Fitness 0.12664389736297477:
(Exp((-Cos(phi))))*(((Exp((-Cos(phi))))^(Tan(&G000017750147449794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Cos(phi))
(Exp((-Cos(phi))))*(((Exp((-Cos(phi))))^(Tan(127401.52449295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Cos(phi))

Generation 155, Farm 36, MDD 54, Fitness 0.068203691291804022:
(Exp((-Tan(Cos(phi)))))*(Cos(phi)):((Exp((-Cos(phi))))*(Tan(phi)))*(Cos(phi))
(Exp((-Tan(Cos(phi)))))*(Cos(phi)):((Exp((-Cos(phi))))*(Tan(phi)))*(Cos(phi))

Generation 172, Farm 36, MDD 71, Fitness 0.035659335793247787:
(Exp((-Tan(Cos(phi)))))*(Cos(phi)):((Exp((-Tan(Cos(phi)))))*(Tan(phi)))*(Cos(phi))
(Exp((-Tan(Cos(phi)))))*(Cos(phi)):((Exp((-Tan(Cos(phi)))))*(Tan(phi)))*(Cos(phi))

Generation 199, Farm 36, MDD 98, Fitness 0.020524840902322459:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(&G000017750147449794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(127401.52449295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
```

Generation 204, Farm 36, MDD 103, Fitness 0.020524816847144293:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(&G0000177501474497l4E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(127401.52449215)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 210, Farm 36, MDD 109, Fitness 0.020523638554861886:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(&G0000177501474457794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(127401.52445295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 211, Farm 36, MDD 110, Fitness 0.020225044746187769:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(&G0000177501486497l4E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(127401.51350784)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 218, Farm 36, MDD 117, Fitness 0.019627863193095819:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(&G0000177501574497l4E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(127401.47550704)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 223, Farm 36, MDD 122, Fitness 0.019627847795876396:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(&G0000177501574494l4E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(127401.47550495)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 230, Farm 36, MDD 129, Fitness 0.019627774812738134:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(&G0000177501574594l4E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos(127401.47549504)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 231, Farm 36, MDD 130, Fitness 0.017452450808343914:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(&G000017750147045794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(127401.52045295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 239, Farm 36, MDD 138, Fitness 0.017451817287217002:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(&G000017750147043194E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(127401.52043895)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 241, Farm 36, MDD 140, Fitness 0.015904907918726816:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(&G000017750157449794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(127401.47550704)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 244, Farm 36, MDD 143, Fitness 0.015266776084170734:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(&G000057750147045794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(527401.52045295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 251, Farm 36, MDD 150, Fitness 0.01513867106039242:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(&G000017750150449794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos(127401.40449295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 255, Farm 36, MDD 154, Fitness 0.015066776084170733:
(Exp((-Cos(phi))))*(((Cos(Cos(Tan(phi))))^(Cos(&G000057750147045794E-08)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(Tan(phi))))^(Cos(527401.52045295)))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 257, Farm 36, MDD 156, Fitness 0.012770000432470074:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^((-Cos(Tan(phi))))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^((-Cos(Tan(phi))))*(Cos(phi))):((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 263, Farm 36, MDD 158, Fitness 0.011522238444388187:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(Tan(phi))))^(Cos(&G000017750197045794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(Tan(phi))))^(Cos(127401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 269, Farm 36, MDD 152, Fitness 0.011485912846591444:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(Tan(phi))))^(Cos(&G000037750197045794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(Tan(phi))))^(Cos(327401.07954704)))))*(Tan(Cos(phi))))
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 278, Farm 36, MDD 143, Fitness 0.010998874765696455:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(phi))))^(Cos(&G000017750197045794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(phi))))^(Cos(127401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 279, Farm 36, MDD 142, Fitness 0.010986630667575572:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(phi))))^(Cos(&G000017750191045794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(phi))))^(Cos(127401.01954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 286, Farm 36, MDD 135, Fitness 0.010986622210761229:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(phi))))^(Cos(&G000017750191075794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos((-Cos(phi))))^(Cos(127401.01925295)))))*(Tan(Cos(phi))))
:((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

```
Generation 287, Farm 36, MDD 134, Fitness 0.010807956281247848:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos(Cos(phi)))^(Cos(&G0000177500970477794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos(Cos(phi)))^(Cos(127400.92047295)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 288, Farm 36, MDD 133, Fitness 0.010786453816546497:
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos(Cos(phi)))^(Cos(&G0000377501970457794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(Tan(phi)))))^(Cos((Cos(Cos(phi)))^(Cos(327401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 289, Farm 36, MDD 132, Fitness 0.010382865688131643:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos((-Cos(phi))))^(Cos(&G0000177501970457794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos((-Cos(phi))))^(Cos(127401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 292, Farm 36, MDD 129, Fitness 0.010367145584289686:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos((-Cos(phi))))^(Cos(&G0000377501970457714E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos((-Cos(phi))))^(Cos(327401.07954784)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 296, Farm 36, MDD 125, Fitness 0.010367145581186529:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos((-Cos(phi))))^(Cos(&G0000377501970457794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos((-Cos(phi))))^(Cos(327401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 300, Farm 36, MDD 121, Fitness 0.010183106584664609:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(&G0000177501970067794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(127401.07993295)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 302, Farm 36, MDD 119, Fitness 0.010182865688131645:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(&G0000177501970457794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(127401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 303, Farm 36, MDD 118, Fitness 0.010182816101776326:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(&G0000177501970567794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(127401.07946704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 304, Farm 36, MDD 117, Fitness 0.010167213209169391:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(&G0000177501900457794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos((-Cos(phi))))^(Cos(127401.00045295)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 307, Farm 36, MDD 114, Fitness 0.010167145581186528:
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos(Cos(phi)))^(Cos(&G0000377501970457794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos((-Cos(phi))))^(Cos((Cos(Cos(phi)))^(Cos(327401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 309, Farm 36, MDD 112, Fitness 0.0099828656881316442:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(&G000017750197045794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(127401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 314, Farm 36, MDD 107, Fitness 0.0099692055569465653:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(&G000017750193056792E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(127401.03946702)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 321, Farm 36, MDD 100, Fitness 0.0099671455813804093:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(&G000037750197045799E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(327401.07954709)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 323, Farm 36, MDD 98, Fitness 0.0099671455811865296:
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(&G000037750197045794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi))))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(327401.07954704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi))))*(Tan(phi)))*(Tan(Cos(phi)))
```

```
Generation 326, Farm 36, MDD 95, Fitness 0.0099671455043081631:
(Exp((-Cos(phi)))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(&G000037750197047794E-08)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi)))*(Tan(phi)))*(Tan(Cos(phi)))
(Exp((-Cos(phi)))*(((Cos(Cos(phi)))^(Cos((Cos(Cos(phi)))^(Cos(327401.07952704)))))*(Tan(Cos(phi)))):
((Exp((-Cos(phi)))*(Tan(phi)))*(Tan(Cos(phi)))

Generation 333, Farm 2, MDD 88, Fitness 0.0094433258536626187:
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh((-(-ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))))):
(ArcSin(ArcSinh(Sin(ArcSin(ArcSinh(Sin(phi)))))))/(ArcSinh(Exp(Tan(Cos(phi)))))
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh((-(-ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))))):
(ArcSin(ArcSinh(Sin(ArcSin(ArcSinh(Sin(phi)))))))/(ArcSinh(Exp(Tan(Cos(phi)))))

Generation 336, Farm 2, MDD 85, Fitness 0.0090433258536626194:
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh(ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))):
(ArcSin(ArcSinh(Sin(ArcSin(ArcSinh(Sin(phi)))))))/(ArcSinh(Exp(Tan(Cos(phi)))))
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh(ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))):
(ArcSin(ArcSinh(Sin(ArcSin(ArcSinh(Sin(phi)))))))/(ArcSinh(Exp(Tan(Cos(phi)))))

Generation 364, Farm 2, MDD 57, Fitness 0.0087441739873828569:
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh(ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))):
(Sin(ArcSin(ArcSinh(Sin(phi)))))/(ArcSinh(Exp(Tan(Cos(phi)))))
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh(ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))):
(Sin(ArcSin(ArcSinh(Sin(phi)))))/(ArcSinh(Exp(Tan(Cos(phi)))))

Generation 366, Farm 2, MDD 55, Fitness 0.0085441739873828582:
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh(ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))):
(ArcSinh(Sin(phi)))/(ArcSinh(Exp(Tan(Cos(phi)))))
(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))/(Exp(ArcSinh(ArcSinh(Tan(ArcSinh(ArcSinh(Tan(Cos(phi))))))))):
(ArcSinh(Sin(phi)))/(ArcSinh(Exp(Tan(Cos(phi)))))

Generation 489, Farm 50, MDD 152, Fitness 0.0076263246222341291:
(ArcSinh(ArcSinh(Tan(Cos(phi)))))/(ArcSinh(Exp(Tan(Cos(phi))))):(Sin(Sin(phi)))/(ArcSinh(Exp(Tan(Cos(phi)))))
(ArcSinh(ArcSinh(Tan(Cos(phi)))))/(ArcSinh(Exp(Tan(Cos(phi))))):(Sin(Sin(phi)))/(ArcSinh(Exp(Tan(Cos(phi)))))

Generation 501, Farm 50, MDD 140, Fitness 0.0065008984915313518:
(ArcSinh(ArcSinh(Tan(Cos(phi)))))/(ArcSinh(Exp(Tan(Cos(phi))))):(ArcSinh(Sin(phi)))/(ArcSinh(Exp(Tan(Cos(phi)))))
(ArcSinh(ArcSinh(Tan(Cos(phi)))))/(ArcSinh(Exp(Tan(Cos(phi))))):(ArcSinh(Sin(phi)))/(ArcSinh(Exp(Tan(Cos(phi)))))

Generation 938, Farm 175, MDD 143, Fitness 0.0059467034771734026:
((ArcCos(Sin((Sin(Cos(phi)))/(Abs(Cos(Cos(Cos(Cos(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Cos(phi)):
((ArcCos(Sin((Sin(Cos(phi)))/(Abs(Cos(Cos(Cos(Cos(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Sin(phi))
((ArcCos(Sin((Sin(Cos(phi)))/(Abs(Cos(Cos(Cos(Cos(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Cos(phi)):
((ArcCos(Sin((Sin(Cos(phi)))/(Abs(Cos(Cos(Cos(Cos(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Sin(phi))

Generation 953, Farm 175, MDD 128, Fitness 0.005750132214765934:
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(Sin(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Cos(phi)):
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(Sin(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Sin(phi))
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(Sin(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Cos(phi)):
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(Sin(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Sin(phi))

Generation 964, Farm 175, MDD 117, Fitness 0.0057084179536228451:
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(phi)))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Cos(phi)):
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(Sin(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Sin(phi))
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(phi)))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Cos(phi)):
((ArcCos(Sin((Sin(Cos(phi)))/(Cos(Cos(Cos(Cos(Sin(phi))))))))/(ArcCos(Sin((Cos(phi))*(Cos(phi)))))*(Sin(phi))
```

13.4.10 Round 4, Simulation 1

(reference values table dump skipped)

```
Generation 1, Farm 43, MDD 30, Fitness 217.6975075181015:
(Max(Abs(t),Pi()))+(&G000000015587176367E-08)
(Max(Abs(t),Pi()))+(145.12126332)

Generation 4, Farm 64, MDD 33, Fitness 184.75523006472332:
(Exp(ArcTanh(Tanh(&G000000000497920478E-08))))+(t)
(Exp(ArcTanh(Tanh(4.92979528))))+(t)

Generation 6, Farm 69, MDD 35, Fitness 174.35234335974783:
(&G000000015510787512E-08)+(t)
(145.80712582)+(t)

Generation 7, Farm 43, MDD 36, Fitness 167.90628755182269:
(t)+(&G000000015587176367E-08)
(t)+(145.12126332)

Generation 9, Farm 43, MDD 38, Fitness 167.90626446342884:
(t)+(&G000000015587176067E-08)
(t)+(145.12126067)
```

```
Generation 10, Farm 50, MDD 39, Fitness 82.747525369504402:
Sqrt(Exp(Sqrt((&G000000009629539138E-08)+(t))))
Sqrt(Exp(Sqrt((93.70569831)+(t))))

Generation 15, Farm 50, MDD 44, Fitness 80.456399915995334:
Sqrt(Exp(Sqrt((&G000000009619539138E-08)+(t))))
Sqrt(Exp(Sqrt((93.89430168)+(t))))

Generation 27, Farm 50, MDD 44, Fitness 80.456341844417054:
Sqrt(Exp(Sqrt((&G000000009619539738E-08)+(t))))
Sqrt(Exp(Sqrt((93.89430768)+(t))))

Generation 28, Farm 50, MDD 43, Fitness 78.695300391436604:
Sqrt(Exp(Sqrt((&G000000009528279920E-08)+(t))))
Sqrt(Exp(Sqrt((94.28270979)+(t))))

Generation 35, Farm 50, MDD 36, Fitness 78.695300131160337:
Sqrt(Exp(Sqrt((&G000000009528279960E-08)+(t))))
Sqrt(Exp(Sqrt((94.28270938999999)+(t))))

Generation 38, Farm 50, MDD 33, Fitness 78.69405269039072:
Sqrt(Exp(Sqrt((&G000000009528079940E-08)+(t))))
Sqrt(Exp(Sqrt((94.28070959)+(t))))

Generation 46, Farm 50, MDD 25, Fitness 78.693508356479228:
Sqrt(Exp(Sqrt((&G000000009523272920E-08)+(t))))
Sqrt(Exp(Sqrt((94.23722979)+(t))))

Generation 50, Farm 50, MDD 21, Fitness 78.687450409242274:
Sqrt(Exp(Sqrt((&G000000009525272920E-08)+(t))))
Sqrt(Exp(Sqrt((94.25722979)+(t))))

Generation 62, Farm 50, MDD 31, Fitness 78.687448396411014:
Sqrt(Exp(Sqrt((&G000000009525278920E-08)+(t))))
Sqrt(Exp(Sqrt((94.25728979)+(t))))

Generation 63, Farm 50, MDD 32, Fitness 78.687432200330704:
Sqrt(Exp(Sqrt((&G000000009525202020E-08)+(t))))
Sqrt(Exp(Sqrt((94.25797978999999)+(t))))

Generation 69, Farm 50, MDD 38, Fitness 78.687429365999563:
Sqrt(Exp(Sqrt((&G000000009525172920E-08)+(t))))
Sqrt(Exp(Sqrt((94.2587702)+(t))))

Generation 75, Farm 50, MDD 44, Fitness 78.687428528866945:
Sqrt(Exp(Sqrt((&G000000009525142920E-08)+(t))))
Sqrt(Exp(Sqrt((94.25842978999999)+(t))))

Generation 81, Farm 50, MDD 50, Fitness 78.687428528002584:
Sqrt(Exp(Sqrt((&G000000009525143920E-08)+(t))))
Sqrt(Exp(Sqrt((94.25843020000001)+(t))))


Generation 88, Farm 50, MDD 43, Fitness 78.687428527855417:
Sqrt(Exp(Sqrt((&G000000009525143927E-08)+(t))))
Sqrt(Exp(Sqrt((94.25843027000001)+(t))))

Generation 89, Farm 50, MDD 42, Fitness 78.68742850907266:
Sqrt(Exp(Sqrt((&G000000009525143020E-08)+(t))))
Sqrt(Exp(Sqrt((94.25843979)+(t))))

Generation 96, Farm 50, MDD 35, Fitness 78.687428465592632:
Sqrt(Exp(Sqrt((&G000000009525147900E-08)+(t))))
Sqrt(Exp(Sqrt((94.25847)+(t))))

Generation 100, Farm 50, MDD 31, Fitness 78.687428449819464:
Sqrt(Exp(Sqrt((&G000000009525149990E-08)+(t))))
Sqrt(Exp(Sqrt((94.25849099)+(t))))

Generation 122, Farm 50, MDD 31, Fitness 78.68742844692072:
Sqrt(Exp(Sqrt((&G000000009525149090E-08)+(t))))
Sqrt(Exp(Sqrt((94.258499)+(t))))

Generation 126, Farm 50, MDD 35, Fitness 78.687428445745127:
Sqrt(Exp(Sqrt((&G000000009525159909E-08)+(t))))
Sqrt(Exp(Sqrt((94.25850990000001)+(t))))

Generation 129, Farm 50, MDD 38, Fitness 78.6874284457393:
Sqrt(Exp(Sqrt((&G000000009525159925E-08)+(t))))
Sqrt(Exp(Sqrt((94.25850973999999)+(t))))

Generation 132, Farm 50, MDD 41, Fitness 78.687428445721295:
Sqrt(Exp(Sqrt((&G000000009525159990E-08)+(t))))
Sqrt(Exp(Sqrt((94.258509)+(t))))
```

```
Generation 180, Farm 50, MDD 29, Fitness 78.68742844572111:
Sqrt(Exp(Sqrt((&G0000000009525159890E-08)+(t))))
Sqrt(Exp(Sqrt((94.25850899)+(t))))

Generation 187, Farm 50, MDD 36, Fitness 78.687428445719334:
Sqrt(Exp(Sqrt((&G0000000009525159809E-08)+(t))))
Sqrt(Exp(Sqrt((94.25850809000001)+(t))))

Generation 190, Farm 50, MDD 39, Fitness 78.687428445718439:
Sqrt(Exp(Sqrt((&G0000000009525159810E-08)+(t))))
Sqrt(Exp(Sqrt((94.25850819)+(t))))

Generation 199, Farm 50, MDD 48, Fitness 78.687428445717444:
Sqrt(Exp(Sqrt((&G0000000009525159859E-08)+(t))))
Sqrt(Exp(Sqrt((94.2585085)+(t))))

Generation 216, Farm 50, MDD 35, Fitness 78.687428445717416:
Sqrt(Exp(Sqrt((&G0000000009525159849E-08)+(t))))
Sqrt(Exp(Sqrt((94.25850849)+(t))))

Generation 236, Farm 50, MDD 25, Fitness 78.687428445717387:
Sqrt(Exp(Sqrt((&G0000000009525159846E-08)+(t))))
Sqrt(Exp(Sqrt((94.25850846)+(t))))

Generation 360, Farm 60, MDD 29, Fitness 78.183660387661149:
((t)*(&G0000000000339357845E-08))+(&G000000017718958000E-08)
((t)*(3.69652845))+(127.88948)

Generation 376, Farm 60, MDD 45, Fitness 77.960063946600712:
((t)*(&G0000000000339357845E-08))+(&G000000017798958000E-08)
((t)*(3.69652845))+(127.08948)

Generation 377, Farm 60, MDD 46, Fitness 77.89596305686581:
((t)*(&G0000000000339357845E-08))+(&G000000017778958000E-08)
((t)*(3.69652845))+(127.28948)

Generation 378, Farm 60, MDD 47, Fitness 77.608010508284693:
((t)*(&G0000000000309357845E-08))+(&G000000017718958000E-08)
((t)*(3.90347154))+(127.88948)

Generation 380, Farm 60, MDD 49, Fitness 77.607754193541822:
((t)*(&G0000000000309350845E-08))+(&G000000017718958000E-08)
((t)*(3.90340845))+(127.88948)

Generation 382, Farm 60, MDD 49, Fitness 77.552741792000305:
((t)*(&G0000000000319357885E-08))+(&G000000017718018000E-08)
((t)*(3.89652885))+(127.88011999)

Generation 385, Farm 60, MDD 46, Fitness 77.363324164538597:
((t)*(&G0000000000309357845E-08))+(&G000000017710958000E-08)
((t)*(3.90347154))+(127.80948)

Generation 390, Farm 60, MDD 41, Fitness 76.583501861743386:
((t)*(&G0000000000319357885E-08))+(&G000000017748018000E-08)
((t)*(3.89652885))+(127.51988)

Generation 396, Farm 60, MDD 35, Fitness 75.770194118920955:
((t)*(&G0000000000309350845E-08))+(&G000000017788958000E-08)
((t)*(3.90340845))+(127.11051999)

Generation 397, Farm 60, MDD 34, Fitness 75.189302168197301:
((t)*(&G0000000000309357845E-08))+(&G000000017618958000E-08)
((t)*(3.90347154))+(126.11051999)

Generation 398, Farm 60, MDD 33, Fitness 75.189302164003891:
((t)*(&G0000000000309357845E-08))+(&G000000017618957000E-08)
((t)*(3.90347154))+(126.11052)

Generation 406, Farm 60, MDD 25, Fitness 75.189302130456639:
((t)*(&G0000000000309357845E-08))+(&G000000017618957008E-08)
((t)*(3.90347154))+(126.11052008)

Generation 409, Farm 60, MDD 22, Fitness 75.108677895410608:
((t)*(&G0000000000309957845E-08))+(&G000000017618957000E-08)
((t)*(3.90947154))+(126.11052)

Generation 415, Farm 60, MDD 24, Fitness 74.774694654491995:
((t)*(&G0000000000306357845E-08))+(&G000000017618957008E-08)
((t)*(3.93652845))+(126.11052008)

Generation 416, Farm 60, MDD 25, Fitness 74.310167716093517:
((t)*(&G0000000000301357845E-08))+(&G000000017618957000E-08)
((t)*(3.98347154))+(126.11052)
```

```
Generation 433, Farm 60, MDD 42, Fitness 73.789864660693183:
((t)*(&G000000000409357845E-08))+(&G000000017618957000E-08)
((t)*(4.09652845))+(126.11052)

Generation 438, Farm 60, MDD 47, Fitness 73.789810422204241:
((t)*(&G000000000409350845E-08))+(&G000000017618957000E-08)
((t)*(4.09659154))+(126.11052)

Generation 440, Farm 60, MDD 49, Fitness 73.787614906623475:
((t)*(&G000000000409057845E-08))+(&G000000017618957000E-08)
((t)*(4.09947154))+(126.11052)

Generation 442, Farm 60, MDD 49, Fitness 73.78761489995297:
((t)*(&G000000000409057844E-08))+(&G000000017618957000E-08)
((t)*(4.09947155))+(126.11052)

Generation 444, Farm 60, MDD 47, Fitness 73.75584829373048:
((t)*(&G000000000409057845E-08))+(&G000000017608957000E-08)
((t)*(4.09947154))+(126.08947999)

Generation 456, Farm 60, MDD 35, Fitness 73.755818518544928:
((t)*(&G000000000409057845E-08))+(&G000000017608955000E-08)
((t)*(4.09947154))+(126.08945999)

Generation 457, Farm 60, MDD 34, Fitness 73.749909176623177:
((t)*(&G000000000409057845E-08))+(&G000000017608557000E-08)
((t)*(4.09947154))+(126.08547999)

Generation 465, Farm 60, MDD 26, Fitness 73.498848588539801:
((t)*(&G000000000409307845E-08))+(&G000000017518057000E-08)
((t)*(4.09607154))+(125.88052)

Generation 476, Farm 60, MDD 25, Fitness 73.498092438365376:
((t)*(&G000000000409057845E-08))+(&G000000017518957000E-08)
((t)*(4.09947154))+(125.88947999)

Generation 483, Farm 60, MDD 32, Fitness 73.244036869809577:
((t)*(&G000000000409307845E-08))+(&G000000017558057000E-08)
((t)*(4.09607154))+(125.48052)

Generation 494, Farm 60, MDD 43, Fitness 73.2317995542993:
((t)*(&G000000000409307845E-08))+(&G000000017568057000E-08)
((t)*(4.09607154))+(125.31947999)

Generation 498, Farm 60, MDD 47, Fitness 73.202927469163043:
((t)*(&G000000000409057845E-08))+(&G000000017568957000E-08)
((t)*(4.09947154))+(125.31052)

Generation 501, Farm 60, MDD 50, Fitness 73.202388250574813:
((t)*(&G000000000409057845E-08))+(&G000000017568057000E-08)
((t)*(4.09947154))+(125.31947999)

Generation 506, Farm 60, MDD 45, Fitness 73.195522552628191:
((t)*(&G000000000419057845E-08))+(&G000000017578957002E-08)
((t)*(4.10052845))+(125.28947997)

Generation 514, Farm 60, MDD 37, Fitness 73.195495464564729:
((t)*(&G000000000419056845E-08))+(&G000000017578957002E-08)
((t)*(4.10053154))+(125.28947997)

Generation 515, Farm 60, MDD 36, Fitness 73.17000861890493:
((t)*(&G000000000419357845E-08))+(&G000000017578957002E-08)
((t)*(4.10347154))+(125.28947997)

Generation 520, Farm 60, MDD 31, Fitness 73.164992534353757:
((t)*(&G000000000419307848E-08))+(&G000000017568057000E-08)
((t)*(4.10392848))+(125.31947999)

Generation 526, Farm 60, MDD 25, Fitness 73.13563917804764:
((t)*(&G000000000429307845E-08))+(&G000000017558067000E-08)
((t)*(4.29607154))+(125.48067999)

Generation 527, Farm 33, MDD 24, Fitness 72.893750240122017:
((&G000000000412474196E-08)*(t))+(&G000000017558956405E-08)
((4.17524106)*(t))+(125.48946405)

Generation 528, Farm 33, MDD 23, Fitness 72.642245797282058:
((&G000000000412474196E-08)*(t))+(&G000000017588956405E-08)
((4.17524106)*(t))+(125.11053594)

Generation 537, Farm 33, MDD 26, Fitness 72.632235509375278:
((&G000000000412474196E-08)*(t))+(&G000000017596956405E-08)
((4.17524106)*(t))+(125.06946405)
```

```
Generation 543, Farm 60, MDD 32, Fitness 72.461345308020256:
((t)*(&G000000000429307845E-08))+(&G000000017588067000E-08)
((t)*(4.29607154))+(125.11932)

Generation 545, Farm 33, MDD 34, Fitness 72.445898754838254:
((&G000000000422474196E-08)*(t))+(&G000000017588956405E-08)
((4.22475893)*(t))+(125.11053594)

Generation 555, Farm 33, MDD 44, Fitness 72.437120340366036:
((&G000000000422874196E-08)*(t))+(&G000000017588956405E-08)
((4.22875893)*(t))+(125.11053594)

Generation 556, Farm 60, MDD 45, Fitness 72.008351785708967:
((t)*(&G000000000429057845E-08))+(&G000000017478957002E-08)
((t)*(4.29947154))+(124.71052002)

Generation 559, Farm 60, MDD 48, Fitness 72.008034765201799:
((t)*(&G000000000429057845E-08))+(&G000000017478907002E-08)
((t)*(4.29947154))+(124.71007997)

Generation 562, Farm 60, MDD 49, Fitness 71.918567729924931:
((t)*(&G000000000429307845E-08))+(&G000000017458067000E-08)
((t)*(4.29607154))+(124.51932)

Generation 566, Farm 60, MDD 45, Fitness 71.918506844065561:
((t)*(&G000000000429307845E-08))+(&G000000017458087000E-08)
((t)*(4.29607154))+(124.51912)

Generation 583, Farm 60, MDD 28, Fitness 71.911653278867803:
((t)*(&G000000000429107845E-08))+(&G000000017458087000E-08)
((t)*(4.29807154))+(124.51912)

Generation 585, Farm 0, MDD 26, Fitness 71.53089371926572:
(&G000000017418956107E-08)+((t)*(&G000000000441398503E-08))
(124.11053807)+((t)*(4.41691403))

Generation 600, Farm 0, MDD 29, Fitness 71.427347826959235:
(&G000000017318956107E-08)+((t)*(&G000000000441398503E-08))
(123.88946192)+((t)*(4.41691403))

Generation 611, Farm 0, MDD 40, Fitness 71.427344302686492:
(&G000000017318956107E-08)+((t)*(&G000000000441398303E-08))
(123.88946192)+((t)*(4.41691603))

Generation 615, Farm 0, MDD 44, Fitness 71.426154423280153:
(&G000000017318456107E-08)+((t)*(&G000000000441398503E-08))
(123.88453807)+((t)*(4.41691403))

Generation 619, Farm 0, MDD 48, Fitness 71.403891322060019:
(&G000000017318956107E-08)+((t)*(&G000000000444391503E-08))
(123.88946192)+((t)*(4.44301403))

Generation 622, Farm 0, MDD 49, Fitness 71.328210134835416:
(&G000000017218956107E-08)+((t)*(&G000000000467398503E-08))
(122.11053807)+((t)*(4.67691403))

Generation 624, Farm 0, MDD 47, Fitness 71.328209700704704:
(&G000000017218956167E-08)+((t)*(&G000000000467398503E-08))
(122.11053867)+((t)*(4.67691403))

Generation 626, Farm 0, MDD 45, Fitness 71.315527938114855:
(&G000000017218956167E-08)+((t)*(&G000000000468398503E-08))
(122.11053867)+((t)*(4.68308596))

Generation 640, Farm 0, MDD 31, Fitness 71.315524237131797:
(&G000000017218956167E-08)+((t)*(&G000000000468398703E-08))
(122.11053867)+((t)*(4.68308796))

Generation 642, Farm 0, MDD 29, Fitness 71.288396665444239:
(&G000000017212956167E-08)+((t)*(&G000000000467398503E-08))
(122.17053867)+((t)*(4.67691403))

Generation 652, Farm 0, MDD 21, Fitness 71.223099961279971:
(&G000000017238956167E-08)+((t)*(&G000000000468398503E-08))
(122.31053867)+((t)*(4.68308596))

Generation 656, Farm 0, MDD 25, Fitness 71.197700963365733:
(&G000000017248956107E-08)+((t)*(&G000000000467393503E-08))
(122.48946192)+((t)*(4.67696596))

Generation 665, Farm 0, MDD 34, Fitness 71.197607725397773:
(&G000000017248656107E-08)+((t)*(&G000000000467393503E-08))
(122.48653807)+((t)*(4.67696596))
```

```
Generation 677, Farm 0, MDD 46, Fitness 71.189879517602549:
(&G000000017248956107E-08)+((t)*(&G000000000467893503E-08))
(122.48946192)+((t)*(4.67103403))

Generation 679, Farm 58, MDD 48, Fitness 71.143755114629087:
((t)*(&G000000000450988981E-08))+(&G000000017289956200E-08)
((t)*(4.59088918))+(122.89053799)

Generation 693, Farm 58, MDD 38, Fitness 71.143674119398042:
((t)*(&G000000000450988981E-08))+(&G000000017289356200E-08)
((t)*(4.59088918))+(122.89653799)

Generation 706, Farm 58, MDD 25, Fitness 71.143507459821734:
((t)*(&G000000000450788981E-08))+(&G000000017289956200E-08)
((t)*(4.59288918))+(122.89053799)

Generation 720, Farm 58, MDD 29, Fitness 71.143507459793682:
((t)*(&G000000000450788981E-08))+(&G000000017289956207E-08)
((t)*(4.59288918))+(122.89053792)

Generation 727, Farm 58, MDD 36, Fitness 71.143507446561699:
((t)*(&G000000000450788951E-08))+(&G000000017289956200E-08)
((t)*(4.59288941))+(122.89053799)

Generation 730, Farm 58, MDD 39, Fitness 71.143507423480628:
((t)*(&G000000000450788911E-08))+(&G000000017289956207E-08)
((t)*(4.59288981))+(122.89053792)

Generation 736, Farm 58, MDD 45, Fitness 71.143507419662768:
((t)*(&G000000000450788981E-08))+(&G000000017289936200E-08)
((t)*(4.59288918))+(122.89033799)

Generation 738, Farm 58, MDD 47, Fitness 71.143506228116479:
((t)*(&G000000000450798981E-08))+(&G000000017289956200E-08)
((t)*(4.59291081))+(122.89053799)

Generation 739, Farm 58, MDD 48, Fitness 71.143496305726345:
((t)*(&G000000000450688981E-08))+(&G000000017289956200E-08)
((t)*(4.59311081))+(122.89053799)

Generation 744, Farm 58, MDD 47, Fitness 71.143496285149439:
((t)*(&G000000000450688981E-08))+(&G000000017289956907E-08)
((t)*(4.59311081))+(122.89053007)

Generation 755, Farm 58, MDD 36, Fitness 71.143496012110475:
((t)*(&G000000000450688281E-08))+(&G000000017289956907E-08)
((t)*(4.59311718))+(122.89053007)

Generation 761, Farm 58, MDD 30, Fitness 71.143495505044314:
((t)*(&G000000000450687981E-08))+(&G000000017289956907E-08)
((t)*(4.59312918))+(122.89053007)

Generation 765, Farm 58, MDD 26, Fitness 71.143482505498781:
((t)*(&G000000000450638981E-08))+(&G000000017289956200E-08)
((t)*(4.59368918))+(122.89053799)

Generation 774, Farm 58, MDD 23, Fitness 71.143482350896647:
((t)*(&G000000000450628981E-08))+(&G000000017289956907E-08)
((t)*(4.59371081))+(122.89053007)

Generation 775, Farm 0, MDD 24, Fitness 71.143402536977916:
(&G000000017286916167E-08)+((t)*(&G000000000450398003E-08))
(122.86986132)+((t)*(4.59691996))

Generation 790, Farm 0, MDD 39, Fitness 71.143402535657657:
(&G000000017286916167E-08)+((t)*(&G000000000450398043E-08))
(122.86986132)+((t)*(4.59691956))

Generation 797, Farm 0, MDD 46, Fitness 71.143402530705387:
(&G000000017286916167E-08)+((t)*(&G000000000450398103E-08))
(122.86986132)+((t)*(4.59691803))

Generation 798, Farm 0, MDD 47, Fitness 71.143402512779417:
(&G000000017286916167E-08)+((t)*(&G000000000450398703E-08))
(122.86986132)+((t)*(4.59691203))

Generation 799, Farm 0, MDD 48, Fitness 71.143402477500857:
(&G000000017286906167E-08)+((t)*(&G000000000450398003E-08))
(122.86993867)+((t)*(4.59691996))

Generation 800, Farm 0, MDD 49, Fitness 71.143402477378373:
(&G000000017286906167E-08)+((t)*(&G000000000450398006E-08))
(122.86993867)+((t)*(4.59691993))
```

```
Generation 805, Farm 0, MDD 46, Fitness 71.143402365288964:
(&G0000000017287910167E-08)+((t)*(&G00000000450398003E-08))
(122.87019867)+((t)*(4.59691996))

Generation 815, Farm 0, MDD 36, Fitness 71.143402365222258:
(&G0000000017287910167E-08)+((t)*(&G00000000450398004E-08))
(122.87019867)+((t)*(4.59691995))

Generation 820, Farm 0, MDD 31, Fitness 71.14340232689861:
(&G0000000017287910166E-08)+((t)*(&G00000000450398503E-08))
(122.87019866)+((t)*(4.59691403))

Generation 825, Farm 0, MDD 26, Fitness 71.143402271280223:
(&G0000000017287916167E-08)+((t)*(&G00000000450399003E-08))
(122.87013867)+((t)*(4.59690003))

Generation 834, Farm 0, MDD 23, Fitness 71.143402245540486:
(&G0000000017287910167E-08)+((t)*(&G00000000450399004E-08))
(122.87019867)+((t)*(4.59690004))

Generation 837, Farm 0, MDD 26, Fitness 71.143402245486968:
(&G0000000017287910167E-08)+((t)*(&G00000000450399003E-08))
(122.87019867)+((t)*(4.59690003))

Generation 838, Farm 0, MDD 27, Fitness 71.143402217320997:
(&G0000000017287946167E-08)+((t)*(&G00000000450399003E-08))
(122.87046132)+((t)*(4.59690003))

Generation 847, Farm 0, MDD 36, Fitness 71.143402213302693:
(&G0000000017287936167E-08)+((t)*(&G00000000450399003E-08))
(122.87033867)+((t)*(4.59690003))

Generation 858, Farm 0, MDD 47, Fitness 71.143401841505778:
(&G0000000017287516167E-08)+((t)*(&G00000000450319003E-08))
(122.87413867)+((t)*(4.59610003))

Generation 877, Farm 0, MDD 34, Fitness 71.143401457854054:
(&G0000000017287516167E-08)+((t)*(&G00000000450313003E-08))
(122.87413867)+((t)*(4.59616003))

Generation 882, Farm 0, MDD 29, Fitness 71.143401382904145:
(&G0000000017287516167E-08)+((t)*(&G00000000450311003E-08))
(122.87413867)+((t)*(4.59618003))

Generation 893, Farm 0, MDD 22, Fitness 71.143401382897224:
(&G0000000017287516165E-08)+((t)*(&G00000000450311003E-08))
(122.87413865)+((t)*(4.59618003))

Generation 898, Farm 0, MDD 27, Fitness 71.143401377064436:
(&G0000000017287516165E-08)+((t)*(&G00000000450311103E-08))
(122.87413865)+((t)*(4.59618196))

Generation 899, Farm 0, MDD 28, Fitness 71.143401360514062:
(&G0000000017287516167E-08)+((t)*(&G00000000450311703E-08))
(122.87413867)+((t)*(4.59618796))

Generation 903, Farm 0, MDD 32, Fitness 71.143401360334948:
(&G0000000017287516167E-08)+((t)*(&G00000000450311803E-08))
(122.87413867)+((t)*(4.59618803))

Generation 910, Farm 0, MDD 39, Fitness 71.143401299853167:
(&G0000000017287517165E-08)+((t)*(&G00000000450321003E-08))
(122.87412134)+((t)*(4.59621996))

Generation 923, Farm 0, MDD 48, Fitness 71.143401299364598:
(&G0000000017287517065E-08)+((t)*(&G00000000450321003E-08))
(122.87412065)+((t)*(4.59621996))

Generation 924, Farm 0, MDD 47, Fitness 71.143401297953858:
(&G0000000017287518165E-08)+((t)*(&G00000000450321003E-08))
(122.87411865)+((t)*(4.59621996))

Generation 935, Farm 0, MDD 36, Fitness 71.14340128607806:
(&G0000000017287519165E-08)+((t)*(&G00000000450321003E-08))
(122.87410134)+((t)*(4.59621996))

Generation 945, Farm 0, MDD 26, Fitness 71.143401285617088:
(&G0000000017287519065E-08)+((t)*(&G00000000450321003E-08))
(122.87410065)+((t)*(4.59621996))

Generation 956, Farm 0, MDD 25, Fitness 71.14340118735177:
(&G0000000017287618165E-08)+((t)*(&G00000000450321003E-08))
(122.87388134)+((t)*(4.59621996))
```

```
Generation 974, Farm 0, MDD 43, Fitness 71.143401187342633:
(&G000000017287618169E-08)+((t)*(&G000000000450321003E-08))
(122.8738813)+((t)*(4.59621996))

Generation 976, Farm 0, MDD 45, Fitness 71.143401176603518:
(&G000000017287628165E-08)+((t)*(&G000000000450321003E-08))
(122.87371865)+((t)*(4.59621996))

Generation 978, Farm 0, MDD 47, Fitness 71.14340112620215:
(&G000000017287618165E-08)+((t)*(&G000000000450327003E-08))
(122.87388134)+((t)*(4.59627996))

Generation 982, Farm 0, MDD 49, Fitness 71.143400978177638:
(&G000000017287668165E-08)+((t)*(&G000000000450327003E-08))
(122.87331865)+((t)*(4.59627996))

Generation 1003, Farm 0, MDD 28, Fitness 71.14340096380019:
(&G000000017287662165E-08)+((t)*(&G000000000450327003E-08))
(122.87337865)+((t)*(4.59627996))

Generation 1009, Farm 58, MDD 22, Fitness 71.143400889496803:
((t)*(&G000000000450361981E-08))+(&G000000017287706907E-08)
((t)*(4.59661081))+(122.87206992)

Generation 1017, Farm 0, MDD 26, Fitness 71.143400845665639:
(&G000000017287668175E-08)+((t)*(&G000000000450337003E-08))
(122.87331874)+((t)*(4.59632003))

Generation 1019, Farm 58, MDD 28, Fitness 71.143400745603699:
((t)*(&G000000000450353981E-08))+(&G000000017287706907E-08)
((t)*(4.59656918))+(122.87206992)

Generation 1026, Farm 58, MDD 35, Fitness 71.14340074516609:
((t)*(&G000000000450353981E-08))+(&G000000017287706407E-08)
((t)*(4.59656918))+(122.87206407)

Generation 1038, Farm 58, MDD 47, Fitness 71.143400715369708:
((t)*(&G000000000450357981E-08))+(&G000000017287706907E-08)
((t)*(4.59652918))+(122.87206992)

Generation 1045, Farm 58, MDD 46, Fitness 71.143400709403394:
((t)*(&G000000000450357981E-08))+(&G000000017287708900E-08)
((t)*(4.59652918))+(122.87208999)

Generation 1070, Farm 58, MDD 21, Fitness 71.143400709331161:
((t)*(&G000000000450357961E-08))+(&G000000017287708900E-08)
((t)*(4.59652938))+(122.87208999)

Generation 1074, Farm 58, MDD 23, Fitness 71.143400696819882:
((t)*(&G000000000450357981E-08))+(&G000000017287738900E-08)
((t)*(4.59652918))+(122.87231)

Generation 1080, Farm 58, MDD 29, Fitness 71.143400693962363:
((t)*(&G000000000450357981E-08))+(&G000000017287728900E-08)
((t)*(4.59652918))+(122.87228999)

Generation 1088, Farm 58, MDD 37, Fitness 71.143400690064041:
((t)*(&G000000000450357581E-08))+(&G000000017287738900E-08)
((t)*(4.59652518))+(122.87231)

Generation 1098, Farm 58, MDD 47, Fitness 71.143400674751931:
((t)*(&G000000000450359981E-08))+(&G000000017287728900E-08)
((t)*(4.59650918))+(122.87228999)

Generation 1106, Farm 58, MDD 45, Fitness 71.143400674728227:
((t)*(&G000000000450359989E-08))+(&G000000017287728900E-08)
((t)*(4.5965091))+(122.87228999)

Generation 1117, Farm 58, MDD 34, Fitness 71.143400672165015:
((t)*(&G000000000450359581E-08))+(&G000000017287738900E-08)
((t)*(4.59650518))+(122.87231)

Generation 1129, Farm 23, MDD 22, Fitness 71.143400662843007:
(&G000000017287763066E-08)+((t)*(&G000000000450344889E-08))
(122.87263933)+((t)*(4.59644889))

Generation 1148, Farm 23, MDD 37, Fitness 71.14340066284278:
(&G000000017287763067E-08)+((t)*(&G000000000450344889E-08))
(122.87263932)+((t)*(4.59644889))

Generation 1155, Farm 23, MDD 44, Fitness 71.143400662840861:
(&G000000017287763066E-08)+((t)*(&G000000000450344899E-08))
(122.87263933)+((t)*(4.5964489))
```

Generation 1157, Farm 23, MDD 46, Fitness 71.143400662743474:
(&G000000017287763666E-08)+((t)*(&G00000000450344889E-08))
(122.87263333)+((t)*(4.59644889))

Generation 1159, Farm 23, MDD 48, Fitness 71.143400662726293:
(&G000000017287763866E-08)+((t)*(&G00000000450344889E-08))
(122.87263133)+((t)*(4.59644889))

Generation 1165, Farm 23, MDD 46, Fitness 71.143400662665996:
(&G000000017287763866E-08)+((t)*(&G00000000450344989E-08))
(122.87263133)+((t)*(4.5964491))

Generation 1166, Farm 23, MDD 45, Fitness 71.143400662530638:
(&G000000017287763066E-08)+((t)*(&G00000000450345889E-08))
(122.87263933)+((t)*(4.5964511))

Generation 1178, Farm 23, MDD 33, Fitness 71.143400662518303:
(&G000000017287763066E-08)+((t)*(&G00000000450345869E-08))
(122.87263933)+((t)*(4.5964513))

Generation 1187, Farm 58, MDD 24, Fitness 71.14340066091988:
((t)*(&G00000000450346989E-08))+(&G000000017287748900E-08)
((t)*(4.5964691))+(122.87248999)

Generation 1188, Farm 58, MDD 23, Fitness 71.143400660378148:
((t)*(&G00000000450347989E-08))+(&G000000017287748904E-08)
((t)*(4.59647089))+(122.87248995)

Generation 1212, Farm 58, MDD 41, Fitness 71.143400660376429:
((t)*(&G00000000450347989E-08))+(&G000000017287748901E-08)
((t)*(4.59647089))+(122.87248998)

Generation 1218, Farm 58, MDD 47, Fitness 71.143400660131888:
((t)*(&G00000000450347289E-08))+(&G000000017287748904E-08)
((t)*(4.5964771))+(122.87248995)

Generation 1219, Farm 58, MDD 48, Fitness 71.143400659631183:
((t)*(&G00000000450347989E-08))+(&G000000017287758904E-08)
((t)*(4.59647089))+(122.87251004)

Generation 1237, Farm 58, MDD 34, Fitness 71.143400659616461:
((t)*(&G00000000450347989E-08))+(&G000000017287758994E-08)
((t)*(4.59647089))+(122.87251095)

Generation 1244, Farm 58, MDD 27, Fitness 71.143400659565103:
((t)*(&G00000000450347989E-08))+(&G000000017287758404E-08)
((t)*(4.59647089))+(122.87251595)

Generation 1248, Farm 58, MDD 23, Fitness 71.143400659558566:
((t)*(&G00000000450347989E-08))+(&G000000017287758204E-08)
((t)*(4.59647089))+(122.87251795)

Generation 1250, Farm 58, MDD 21, Fitness 71.143400659558452:
((t)*(&G00000000450347989E-08))+(&G000000017287758104E-08)
((t)*(4.59647089))+(122.87251804)

Generation 1266, Farm 58, MDD 35, Fitness 71.143400659558409:
((t)*(&G00000000450347989E-08))+(&G000000017287758108E-08)
((t)*(4.59647089))+(122.87251808)

Generation 1267, Farm 58, MDD 36, Fitness 71.143400659558154:
((t)*(&G00000000450347989E-08))+(&G000000017287758154E-08)
((t)*(4.59647089))+(122.87251855)

Generation 1270, Farm 58, MDD 39, Fitness 71.143400659553123:
((t)*(&G00000000450347979E-08))+(&G000000017287758104E-08)
((t)*(4.5964707))+(122.87251804)

Generation 1278, Farm 58, MDD 47, Fitness 71.143400659544227:
((t)*(&G00000000450347959E-08))+(&G000000017287758004E-08)
((t)*(4.5964705))+(122.87251995)

Generation 1280, Farm 58, MDD 49, Fitness 71.143400659539893:
((t)*(&G00000000450347929E-08))+(&G000000017287758004E-08)
((t)*(4.59647029))+(122.87251995)

Generation 1299, Farm 58, MDD 32, Fitness 71.14340065953192:
((t)*(&G00000000450347909E-08))+(&G000000017287757204E-08)
((t)*(4.59647009))+(122.87252204)

Generation 1319, Farm 58, MDD 28, Fitness 71.143400659531125:
((t)*(&G00000000450347905E-08))+(&G000000017287757204E-08)
((t)*(4.59647005))+(122.87252204)

```
Generation 1323, Farm 58, MDD 32, Fitness 71.143400659530585:
((t)*(&G0000000000450347902E-08))+(&G000000017287757204E-08)
((t)*(4.59647002))+(122.87252204)

Generation 1335, Farm 58, MDD 44, Fitness 71.143400659530414:
((t)*(&G0000000000450347902E-08))+(&G000000017287757214E-08)
((t)*(4.59647002))+(122.87252215)

Generation 1338, Farm 58, MDD 47, Fitness 71.143400659529064:
((t)*(&G0000000000450346909E-08))+(&G000000017287757204E-08)
((t)*(4.5964699))+(122.87252204)

Generation 1359, Farm 58, MDD 32, Fitness 71.143400659527487:
((t)*(&G0000000000450346909E-08))+(&G000000017287757274E-08)
((t)*(4.5964699))+(122.87252275)

Generation 1369, Farm 58, MDD 22, Fitness 71.143400659527103:
((t)*(&G0000000000450346909E-08))+(&G000000017287757304E-08)
((t)*(4.5964699))+(122.87252395)

Generation 1387, Farm 58, MDD 36, Fitness 71.143400659521305:
((t)*(&G0000000000450346959E-08))+(&G000000017287757304E-08)
((t)*(4.59646949))+(122.87252395)

Generation 1399, Farm 58, MDD 48, Fitness 71.143400659517766:
((t)*(&G0000000000450346979E-08))+(&G000000017287757704E-08)
((t)*(4.59646929))+(122.87252795)

Generation 1402, Farm 58, MDD 49, Fitness 71.143400659515592:
((t)*(&G0000000000450346970E-08))+(&G000000017287757704E-08)
((t)*(4.5964692))+(122.87252795)

Generation 1404, Farm 35, MDD 47, Fitness 71.143400659511656:
((&G0000000000450346834E-08)*(t))+(&G000000017287756714E-08)
((4.59646835)*(t))+(122.87253215)

Generation 1406, Farm 35, MDD 45, Fitness 71.143400659511627:
((&G0000000000450346814E-08)*(t))+(&G000000017287756714E-08)
((4.59646815)*(t))+(122.87253215)

Generation 1412, Farm 35, MDD 39, Fitness 71.143400659511343:
((&G0000000000450346834E-08)*(t))+(&G000000017287756802E-08)
((4.59646835)*(t))+(122.87253197)

Generation 1413, Farm 35, MDD 38, Fitness 71.143400659511144:
((&G0000000000450346834E-08)*(t))+(&G000000017287756812E-08)
((4.59646835)*(t))+(122.87253182)

Generation 1420, Farm 35, MDD 31, Fitness 71.143400659511087:
((&G0000000000450346834E-08)*(t))+(&G000000017287756822E-08)
((4.59646835)*(t))+(122.87253177)

Generation 1425, Farm 35, MDD 26, Fitness 71.143400659510817:
((&G0000000000450346834E-08)*(t))+(&G000000017287756862E-08)
((4.59646835)*(t))+(122.87253137)

Generation 1436, Farm 35, MDD 25, Fitness 71.143400659510803:
((&G0000000000450346832E-08)*(t))+(&G000000017287756862E-08)
((4.59646837)*(t))+(122.87253137)

Generation 1445, Farm 35, MDD 34, Fitness 71.143400659510775:
((&G0000000000450346833E-08)*(t))+(&G000000017287756872E-08)
((4.59646836)*(t))+(122.87253122)

Generation 1447, Farm 35, MDD 36, Fitness 71.143400659510746:
((&G0000000000450346832E-08)*(t))+(&G000000017287756882E-08)
((4.59646837)*(t))+(122.87253117)

Generation 1453, Farm 35, MDD 42, Fitness 71.143400659510718:
((&G0000000000450346831E-08)*(t))+(&G000000017287756892E-08)
((4.59646838)*(t))+(122.87253102)

Generation 1486, Farm 58, MDD 25, Fitness 71.143400659510675:
((t)*(&G0000000000450346850E-08))+(&G000000017287757904E-08)
((t)*(4.59646859))+(122.87252995)

Generation 1499, Farm 35, MDD 28, Fitness 71.143400659510675:
((&G0000000000450346841E-08)*(t))+(&G000000017287756892E-08)
((4.59646841)*(t))+(122.87253102)

Generation 1515, Farm 58, MDD 44, Fitness 71.143400659510661:
((t)*(&G0000000000450346850E-08))+(&G000000017287757901E-08)
((t)*(4.59646859))+(122.87252998)
```

13.4.11 Round 4, Simulation 2

(reference values table dump skipped)

```
Generation 1, Farm 133, MDD 80, Fitness 5600733.4465240035:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000797951E-08),(ArcSinh(&G000000007760880278E-08))-(Cosh(&G000000000000054893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00707058),(ArcSinh(72.60880271000001))-(Cosh(0.00055103)))))

Generation 3, Farm 133, MDD 82, Fitness 5600733.4465240035:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000797951E-08),(ArcSinh(&G700000007760880278E-08))-(Cosh(&G000000000000054893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00707058),(ArcSinh(7999999927.391197))-(Cosh(0.00055103)))))

Generation 7, Farm 133, MDD 86, Fitness 5600732.4465240035:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000797951E-08),(Pi())-(Cosh(&G000000000000054893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00707058),(Pi())-(Cosh(0.00055103)))))

Generation 8, Farm 133, MDD 87, Fitness 5599911.4050261332:
(Log(Cosh((-t))))/(ArcTanh(Min((-&G000000000000797901E-08),(ArcSinh(&G000000007760880278E-08))-(Cosh(&G000000000000054893E-08)))))
(Log(Cosh((-t))))/(ArcTanh(Min((-0.00707001),(ArcSinh(72.60880271000001))-(Cosh(0.00055103)))))

Generation 9, Farm 158, MDD 88, Fitness 4129607.6764017586:
((-Abs(((-Exp(Max(Log(Log(&G000040085047983l0E-08)),&G000000000000173856E-08))))*(t))))/(ArcSinh(&G000000000005235739E-08))
((-Abs(((-Exp(Max(Log(Log(40085.95291619)),0.00123146))))*(t))))/(ArcSinh(0.0576523))

Generation 15, Farm 109, MDD 94, Fitness 4126495.9555658358:
(Min(&G000000000001133294E-08,(-Abs(t))))*((-((-Cosh((-Log(&G000000000000276745E-08)))))-(Cos(&G000417844919736549E-08))))
(Min(0.01836295,(-Abs(t))))*((-((-Cosh((-Log(0.00273245)))))-(Cos(4128449.89233449))))

Generation 19, Farm 199, MDD 98, Fitness 3570806.3405103432:
(Cosh(Tanh(&G000020000000000058E-08)))+(((-t))*(((Cos(&G000003000000008934E-08))^((-ArcTan(&G000000485848083569E-08))))*(t)))
(Cosh(Tanh(200000.00000051)))+(((-t))*(((Cos(39999.99991035))^((-ArcTan(4851.5191653))))*(t)))

Generation 20, Farm 199, MDD 99, Fitness 2180757.000297857:
(Cosh(Tanh(&G0000200000000000058E-08)))+(((-t))*(((Cos(&G000003000000008934E-08))^((-ArcTan(&G0000000000761844279E-08))))*(t)))
(Cosh(Tanh(200000.00000051)))+(((-t))*(((Cos(39999.99991035))^((-ArcTan(7.3884427))))*(t)))

Generation 23, Farm 199, MDD 98, Fitness 2180756.9620671771:
(Cosh(Tanh(&G000020000000000058E-08)))+(((-t))*(((Cos(&G000003000000008934E-08))^((-ArcTan(&G0000000000761844209E-08))))*(t)))
(Cosh(Tanh(200000.00000051)))+(((-t))*(((Cos(39999.99991035))^((-ArcTan(7.38844209))))*(t)))

Generation 27, Farm 199, MDD 94, Fitness 2098184.0372177344:
(t)+(((-t))*(((Cos(&G000003000000008934E-08))^((-ArcTan(&G0000000000761844209E-08))))*(t)))
(t)+(((-t))*(((Cos(39999.99991035))^((-ArcTan(7.38844209))))*(t)))

Generation 30, Farm 119, MDD 91, Fitness 1593983.05677349:
((ArcSinh(Tan((&G0717825468859702 79E-08)-(&G000108882760078871E-08))))-(t))*((t)/((-Tan((-&G000135299305233777E-08)))))
((ArcSinh(Tan((787174468.8507973)-(1911172.60071121))))-(t))*((t)/((-Tan((-1657096.05763272)))))

Generation 32, Farm 119, MDD 89, Fitness 1562114.4901511469:
((ArcSinh(Tan((&G0710825468859702 79E-08)-(&G000108582760078841E-08))))-(t))*((t)/((-Tan((-&G000135299305233777E-08)))))
((ArcSinh(Tan((780825531.1492027)-(1914827.39928841))))-(t))*((t)/((-Tan((-1657096.05763272)))))

Generation 34, Farm 119, MDD 87, Fitness 523595.07083479478:
((Log(Sqrt(&G0072382823708744444E-08)))-(t))*((t)/((-Tan((-&G000135299305233777E-08)))))
((Log(Sqrt(77682823.20875555)))-(t))*((t)/((-Tan((-1657096.05763272)))))

Generation 37, Farm 119, MDD 84, Fitness 523540.38430735457:
((ArcSinh(&G0000045348291479l E-08))-(t))*((t)/((-Tan((-&G000135299305233777E-08)))))
((ArcSinh(4564.82984701))-(t))*((t)/((-Tan((-1657096.05763272)))))

Generation 50, Farm 119, MDD 71, Fitness 523540.33702124929:
((ArcSinh(&G0000045358291479l E-08))-(t))*((t)/((-Tan((-&G000135299305233777E-08)))))
((ArcSinh(4565.17015298))-(t))*((t)/((-Tan((-1657096.05763272)))))

Generation 52, Farm 119, MDD 69, Fitness 523540.11047261988:
((ArcSinh(&G0000045368291429l E-08))-(t))*((t)/((-Tan((-&G000135299305233777E-08)))))
((ArcSinh(4566.82984298))-(t))*((t)/((-Tan((-1657096.05763272)))))

Generation 53, Farm 119, MDD 68, Fitness 364149.47115667997:
((ArcSinh(&G0000024042932007 40E-08))-(t))*((t)/((-Tan((-&G000135299306233777E-08)))))
((ArcSinh(24042.96200759))-(t))*((t)/((-Tan((-1657096.06236727)))))

Generation 54, Farm 39, MDD 67, Fitness 222099.04329552295:
(((t)*((-Tan((-Sqrt(&G00200017446597 5185E-08))))))+(&G000000003005602838E-08))*(t)
(((t)*((-Tan((-Sqrt(20001244.65074114))))))+(39.94602831))*(t)

Generation 55, Farm 119, MDD 66, Fitness 37407.187265697372:
((ArcSinh(&G0000004544291479l E-08))-(t))*((t)/((-Tan((-&G000000000020000002E-08)))))
((ArcSinh(4555.57015298))-(t))*((t)/((-Tan((-0.20000002)))))
```

```
Generation 59, Farm 119, MDD 62, Fitness 25252.330174095641:
((ArcSinh(&G0000024544429147 91E-08))-(t))*((t)/((-Tan((-&G00000000020000002E-08)))))
((ArcSinh(24555.57015298))-(t))*((t)/((-Tan((-0.20000002)))))

Generation 65, Farm 119, MDD 56, Fitness 25252.330171906273:
((ArcSinh(&G0000024544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020000002E-08)))))
((ArcSinh(24555.57015101))-(t))*((t)/((-Tan((-0.20000002)))))

Generation 74, Farm 119, MDD 47, Fitness 25207.89447490064:
((ArcSinh(&G0000024584429147 91E-08))-(t))*((t)/((-Tan((-&G00000000020000002E-08)))))
((ArcSinh(24515.57015298))-(t))*((t)/((-Tan((-0.20000002)))))

Generation 77, Farm 119, MDD 44, Fitness 25191.156539560423:
((ArcSinh(&G0000024544429147 91E-08))-(t))*((t)/((-Tan((-&G00000000020000702E-08)))))
((ArcSinh(24555.57015298))-(t))*((t)/((-Tan((-0.20000797)))))

Generation 78, Farm 119, MDD 43, Fitness 18191.166296188901:
((ArcSinh(&G0000024544429147 91E-08))-(t))*((t)/((-Tan((-&G00000000020800002E-08)))))
((ArcSinh(24555.57015298))-(t))*((t)/((-Tan((-0.20800002)))))

Generation 86, Farm 119, MDD 35, Fitness 10305.012912127479:
((ArcSinh(&G0000024544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020500002E-08)))))
((ArcSinh(24555.57015101))-(t))*((t)/((-Tan((-0.20599997)))))

Generation 90, Farm 119, MDD 31, Fitness 8604.0254612052286:
((ArcSinh(&G0000024544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020300002E-08)))))
((ArcSinh(24555.57015101))-(t))*((t)/((-Tan((-0.20399997)))))

Generation 96, Farm 119, MDD 35, Fitness 8586.422928053993:
((ArcSinh(&G0000024544429148 91E-08))-(t))*((t)/((-ArcTanh((-&G000000000020300002E-08)))))
((ArcSinh(24555.57015101))-(t))*((t)/((-ArcTanh((-0.20399997)))))

Generation 108, Farm 119, MDD 47, Fitness 6525.7509902854799:
((ArcSinh(&G0000021544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020300002E-08)))))
((ArcSinh(21444.42984898))-(t))*((t)/((-Tan((-0.20399997)))))

Generation 118, Farm 119, MDD 57, Fitness 4483.6114618558795:
((ArcSinh(&G0000014544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020300002E-08)))))
((ArcSinh(15444.42984898))-(t))*((t)/((-Tan((-0.20399997)))))

Generation 124, Farm 119, MDD 63, Fitness 4469.8960052726306:
((ArcSinh(&G0000014544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020308002E-08)))))
((ArcSinh(15444.42984898))-(t))*((t)/((-Tan((-0.20391997)))))

Generation 130, Farm 119, MDD 69, Fitness 4469.4825691400492:
((ArcSinh(&G0000014544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020308402E-08)))))
((ArcSinh(15444.42984898))-(t))*((t)/((-Tan((-0.20391597)))))

Generation 133, Farm 119, MDD 72, Fitness 4468.7955274012165:
((ArcSinh(&G0000014544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020320000E-08)))))
((ArcSinh(15444.42984898))-(t))*((t)/((-Tan((-0.20379999)))))

Generation 135, Farm 119, MDD 74, Fitness 4468.7955273091511:
((ArcSinh(&G0000014544429140 91E-08))-(t))*((t)/((-Tan((-&G00000000020320000E-08)))))
((ArcSinh(15444.42984098))-(t))*((t)/((-Tan((-0.20379999)))))

Generation 139, Farm 119, MDD 78, Fitness 4468.0921533964929:
((ArcSinh(&G0000014544429148 41E-08))-(t))*((t)/((-Tan((-&G00000000020309002E-08)))))
((ArcSinh(15444.42984841))-(t))*((t)/((-Tan((-0.20390002)))))

Generation 144, Farm 119, MDD 83, Fitness 4467.1543501596943:
((ArcSinh(&G0000014544429140 91E-08))-(t))*((t)/((-ArcTanh((-&G000000000020320000E-08)))))
((ArcSinh(15444.42984098))-(t))*((t)/((-ArcTanh((-0.20379999)))))

Generation 149, Farm 119, MDD 88, Fitness 4466.9385245777949:
((ArcSinh(&G0000014544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020318012E-08)))))
((ArcSinh(15444.42984898))-(t))*((t)/((-Tan((-0.20388017)))))

Generation 150, Farm 119, MDD 89, Fitness 4466.9322426936787:
((ArcSinh(&G0000014544429148 91E-08))-(t))*((t)/((-Tan((-&G00000000020318002E-08)))))
((ArcSinh(15444.42984898))-(t))*((t)/((-Tan((-0.20388002)))))

Generation 152, Farm 119, MDD 91, Fitness 4466.9322426592998:
((ArcSinh(&G0000014544429108 91E-08))-(t))*((t)/((-Tan((-&G00000000020318002E-08)))))
((ArcSinh(15444.42980898))-(t))*((t)/((-Tan((-0.20388002)))))

Generation 156, Farm 119, MDD 95, Fitness 4466.9322375045012:
((ArcSinh(&G0000014544423108 91E-08))-(t))*((t)/((-Tan((-&G00000000020318002E-08)))))
((ArcSinh(15444.42380898))-(t))*((t)/((-Tan((-0.20388002)))))

Generation 158, Farm 119, MDD 97, Fitness 4466.9302176954607:
((ArcSinh(&G0000014540829178 91E-08))-(t))*((t)/((-Tan((-&G00000000020318002E-08)))))
((ArcSinh(15440.82987101))-(t))*((t)/((-Tan((-0.20388002)))))
```

Generation 160, Farm 119, MDD 99, Fitness 4466.4826149873952:
((ArcSinh(&G000001454442914891E-08))-(t))*((t)/((-Tan((-&G000000000020314302E-08)))))
((ArcSinh(15444.42984898))-(t))*((t)/((-Tan((-0.20384397)))))

Generation 163, Farm 119, MDD 98, Fitness 4466.4826110410431:
((ArcSinh(&G000001454442984891E-08))-(t))*((t)/((-Tan((-&G000000000020314302E-08)))))
((ArcSinh(15444.42915101))-(t))*((t)/((-Tan((-0.20384397)))))

Generation 167, Farm 39, MDD 94, Fitness 4179.66253175152:
((((t)*((-Tan((-(-&G1000600000000015709E-08))))))+(&G000000005710006808E-08))*(t))+(&G000000008610000006E-08)
((((t)*((-Tan((-(-1999399999.999852))))))+(52.19993191))*(t))+(86.19999993)

Generation 170, Farm 39, MDD 91, Fitness 1517.6981991086761:
((((t)*((-Tan((-(-&G1000600000000016709E-08))))))+(&G000000005910010008E-08))*(t))+(&G000000005610800006E-08)
((((t)*((-Tan((-(-1999399999.999868))))))+(50.19980008))*(t))+(53.80800006)

Generation 176, Farm 105, MDD 85, Fitness 1328.8452107556975:
((t)*((((-t))*(&G000000000485100795E-08))-((-&G000000005901010393E-08))))-((-&G000000005900018293E-08))
((t)*((((-t))*(4.85800705))-((-50.01980303))))-((-50.00011703))

Generation 177, Farm 39, MDD 84, Fitness 1099.2020083582108:
((((t)*((-Tan((-(-&G1000600000000050709E-08))))))+(&G000000005910010008E-08))*(t))+(&G000000005610800006E-08)
((((t)*((-Tan((-(-1999399999.999408))))))+(50.19980008))*(t))+(53.80800006)

Generation 184, Farm 39, MDD 77, Fitness 1065.8941071283357:
((((t)*((-Tan((-(-&G1000600000000050709E-08))))))+(&G000000005900000008E-08))*(t))+(&G000000005610800006E-08)
((((t)*((-Tan((-(-1999399999.999408))))))+(50.00000008))*(t))+(53.80800006)

Generation 185, Farm 39, MDD 76, Fitness 1065.8938492282591:
((((t)*((-Tan((-(-&G1000600000000050709E-08))))))+(&G000000005900010008E-08))*(t))+(&G000000005610800006E-08)
((((t)*((-Tan((-(-1999399999.999408))))))+(50.00019991))*(t))+(53.80800006)

Generation 188, Farm 139, MDD 73, Fitness 818.76850815066337:
(&G000000008805304980E-08)+((t)*((&G000000004930919715E-08)+((t)/(Tan(&G000000034835400093E-08)))))
(88.05604919)+((t)*((49.60989214)+((t)/(Tan(351.65599903)))))

Generation 200, Farm 39, MDD 61, Fitness 122.37963022771682:
((((t)*((-Tan((-(-&G1000600000000016709E-08))))))+(&G000000005910010008E-08))*(t))+(&G000000009612800086E-08)
((((t)*((-Tan((-(-1999399999.999868))))))+(50.19980008))*(t))+(93.82800086)

Generation 216, Farm 39, MDD 45, Fitness 122.17895374391989:
((((t)*((-Tan((-(-&G1000600000000016709E-08))))))+(&G000000005910080008E-08))*(t))+(&G000000009612800086E-08)
((((t)*((-Tan((-(-1999399999.999868))))))+(50.19919991))*(t))+(93.82800086)

Generation 218, Farm 39, MDD 43, Fitness 118.89605419518523:
((((t)*((-Tan((-(-&G1000600000000016709E-08))))))+(&G000000005910010008E-08))*(t))+(&G000000009692800086E-08)
((((t)*((-Tan((-(-1999399999.999868))))))+(50.19980008))*(t))+(93.02800086000001)

Generation 219, Farm 39, MDD 42, Fitness 100.97659990544655:
((((t)*((-Tan((-(-&G1000600000000016709E-08))))))+(&G000000005918010008E-08))*(t))+(&G000000009612800086E-08)
((((t)*((-Tan((-(-1999399999.999868))))))+(50.11980008))*(t))+(93.82800086)

Generation 229, Farm 39, MDD 32, Fitness 70.950542581157649:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005910010008E-08))*(t))+(&G000000009612800086E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.19980008))*(t))+(93.82800086)

Generation 241, Farm 39, MDD 40, Fitness 70.950542462143119:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005910010008E-08))*(t))+(&G000000009612800088E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.19980008))*(t))+(93.82800088)

Generation 245, Farm 39, MDD 44, Fitness 63.147638091041344:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005910010008E-08))*(t))+(&G000000009212800086E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.19980008))*(t))+(97.82800086)

Generation 249, Farm 39, MDD 48, Fitness 49.547512258885035:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005918010008E-08))*(t))+(&G000000009612800086E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.11980008))*(t))+(93.82800086)

Generation 252, Farm 39, MDD 51, Fitness 45.946809404046483:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005919010008E-08))*(t))+(&G000000009612800086E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.10019991))*(t))+(93.82800086)

Generation 259, Farm 39, MDD 58, Fitness 45.946737503504025:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005919010058E-08))*(t))+(&G000000009612800086E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.10019948))*(t))+(93.82800086)

Generation 260, Farm 39, MDD 59, Fitness 38.967186392298203:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005919010008E-08))*(t))+(&G000000009112800086E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.10019991))*(t))+(98.17199913)

Generation 269, Farm 39, MDD 68, Fitness 38.967163755141591:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005919010008E-08))*(t))+(&G000000009112800986E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.10019991))*(t))+(98.17199085999999)

Generation 277, Farm 39, MDD 76, Fitness 37.40515933019617:
((((t)*((-Tan((-(-&G1000500000000036709E-08))))))+(&G000000005919010008E-08))*(t))+(&G000000009112800086E-08)
((((t)*((-Tan((-(-1999400000.000332))))))+(50.10019991))*(t))+(98.17199913)

Generation 279, Farm 39, MDD 78, Fitness 29.839478961088766:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005900010008E-08))*(t))+(&G000000009212800086E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.00019991))*(t))+(97.82800086)

Generation 287, Farm 39, MDD 86, Fitness 29.11294416234627:
((((t)*((-Tan((-(-&G1000500000000016709E-08))))))+(&G000000005900010008E-08))*(t))+(&G000000009262800081E-08)
((((t)*((-Tan((-(-1999400000.000132))))))+(50.00019991))*(t))+(97.37199918)

Generation 289, Farm 39, MDD 88, Fitness 28.109806502632726:
((((t)*((-Tan((-(-&G1000500000000016009E-08))))))+(&G000000005900010008E-08))*(t))+(&G000000009262800081E-08)
((((t)*((-Tan((-(-1999400000.00014))))))+(50.00019991))*(t))+(97.37199918)

Generation 292, Farm 39, MDD 91, Fitness 23.526344008063372:
((((t)*((-Tan((-(-&G1000500000000010709E-08))))))+(&G000000005900010008E-08))*(t))+(&G000000009212800086E-08)
((((t)*((-Tan((-(-1999400000.000192))))))+(50.00019991))*(t))+(97.82800086)

Generation 296, Farm 39, MDD 95, Fitness 23.485866299809345:
((((t)*((-Tan((-(-&G1000500000000010799E-08))))))+(&G000000005900010008E-08))*(t))+(&G000000009212800086E-08)
((((t)*((-Tan((-(-1999400000.000193))))))+(50.00019991))*(t))+(97.82800086)

Generation 297, Farm 39, MDD 96, Fitness 23.485495836441981:
((((t)*((-Tan((-(-&G1000500000000010799E-08))))))+(&G000000005900080008E-08))*(t))+(&G000000009212800086E-08)
((((t)*((-Tan((-(-1999400000.000193))))))+(50.00080008))*(t))+(97.82800086)

Generation 305, Farm 39, MDD 96, Fitness 23.168891498588675:
((((t)*((-Tan((-(-&G1000500000000010099E-08))))))+(&G000000005900080008E-08))*(t))+(&G000000009212800086E-08)
((((t)*((-Tan((-(-1999400000.000199))))))+(50.00080008))*(t))+(97.82800086)

Generation 306, Farm 39, MDD 95, Fitness 23.168845200950322:
((((t)*((-Tan((-(-&G1000500000000010099E-08))))))+(&G000000005900080008E-08))*(t))+(&G000000009212806086E-08)
((((t)*((-Tan((-(-1999400000.000199))))))+(50.00080008))*(t))+(97.82806085999999)

Generation 308, Farm 39, MDD 93, Fitness 19.125997369387942:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900010008E-08))*(t))+(&G000000009212800088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00019991))*(t))+(97.82800088)

Generation 314, Farm 39, MDD 87, Fitness 19.125933863961208:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900010008E-08))*(t))+(&G000000009212807088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00019991))*(t))+(97.82807911)

Generation 318, Farm 39, MDD 83, Fitness 19.125683073200022:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009212800088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(97.82800088)

Generation 326, Farm 39, MDD 75, Fitness 19.092212957094812:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900010008E-08))*(t))+(&G000000009217800088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00019991))*(t))+(97.87199911)

Generation 335, Farm 39, MDD 66, Fitness 19.091898660906896:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009217800088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(97.87199911)

Generation 347, Farm 39, MDD 54, Fitness 19.091898516134904:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009217800068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(97.87199931000001)

Generation 358, Farm 39, MDD 43, Fitness 19.091853698051079:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900090008E-08))*(t))+(&G000000009217800088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00099991))*(t))+(97.87199911)

Generation 359, Farm 39, MDD 42, Fitness 19.090767962522612:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900010008E-08))*(t))+(&G000000009217500088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00019991))*(t))+(97.87400088)

Generation 363, Farm 39, MDD 38, Fitness 19.087590239651664:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009217100088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(97.87800088)

Generation 373, Farm 39, MDD 32, Fitness 19.087585804823039:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009217100788E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(97.87800711)

Generation 377, Farm 39, MDD 36, Fitness 19.078135752874235:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900010008E-08))*(t))+(&G000000009219800088E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00019991))*(t))+(97.89199911)

Generation 379, Farm 39, MDD 38, Fitness 18.972126114276726:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009117800068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.12800068)

Generation 393, Farm 39, MDD 52, Fitness 18.972126087334583:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050002E-08))*(t))+(&G000000009117800068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059997))*(t))+(98.12800068)

Generation 399, Farm 39, MDD 58, Fitness 18.972124709708169:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009117800768E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.12800731)

Generation 403, Farm 39, MDD 62, Fitness 18.963014690017342:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009111800068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.18800068)

Generation 417, Farm 39, MDD 76, Fitness 18.961386180399693:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009121800068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.21199931)

Generation 421, Farm 39, MDD 80, Fitness 18.961386176890219:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009121800060E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.21199939)

Generation 425, Farm 39, MDD 84, Fitness 18.961386176890219:
((((t)*((-Tan(&G100050000000010000E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009121800060E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.21199939)

Generation 431, Farm 39, MDD 90, Fitness 18.96133000971092:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900080008E-08))*(t))+(&G000000009121800060E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00080008))*(t))+(98.21199939)

Generation 434, Farm 39, MDD 93, Fitness 18.961246290409491:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900080008E-08))*(t))+(&G000000009121600060E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00080008))*(t))+(98.21399939)

Generation 437, Farm 39, MDD 96, Fitness 18.961099299682566:
((((t)*((-Tan(&G100050000000010000E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009121000060E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.21999939)

Generation 446, Farm 39, MDD 95, Fitness 18.960940407177127:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009122800068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.22800067999999)

Generation 456, Farm 39, MDD 85, Fitness 18.960940375746159:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050001E-08))*(t))+(&G000000009122800068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059998))*(t))+(98.22800067999999)

Generation 459, Farm 39, MDD 82, Fitness 18.960938091461767:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900050008E-08))*(t))+(&G000000009122810068E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00059991))*(t))+(98.22819930999999)

Generation 466, Farm 39, MDD 75, Fitness 18.960852816691787:
((((t)*((-Tan(&G100050000000010004E-08))))+(&G000000005900080008E-08))*(t))+(&G000000009123800060E-08)
((((t)*((-Tan(1999400000.0002))))+(50.00080008))*(t))+(98.23199939)

Generation 490, Farm 139, MDD 51, Fitness 18.919075278195418:
(&G000000009229912361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835298393E-08)))))
(97.70982338)+((t)*((49.99690709)+((t)/(Tan(351.65708303)))))

Generation 495, Farm 139, MDD 46, Fitness 18.911701376743853:
(&G000000009239912361E-08)+((t)*((&G000000004900209737E-08)+((t)/(Tan(&G000000034835298393E-08)))))
(97.69017660999999)+((t)*((49.99790767)+((t)/(Tan(351.65708303)))))

Generation 498, Farm 139, MDD 43, Fitness 18.756124262984983:
(&G000000009229912361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835298593E-08)))))
(97.70982338)+((t)*((49.99690709)+((t)/(Tan(351.65708503)))))

Generation 499, Farm 139, MDD 42, Fitness 18.177299096009129:
(&G000000009289912341E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835288393E-08)))))
(97.10982358)+((t)*((49.99690709)+((t)/(Tan(351.65711696)))))

Generation 500, Farm 139, MDD 41, Fitness 17.929360347323023:
(&G000000009209952361E-08)+((t)*((&G000000004900999799E-08)+((t)/(Tan(&G000000034835299393E-08)))))
(97.90942338000001)+((t)*((49.99090709)+((t)/(Tan(351.65709696)))))

Generation 509, Farm 139, MDD 32, Fitness 17.018932563024059:
(&G000000009229912361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835288393E-08)))))
(97.70982338)+((t)*((49.99690709)+((t)/(Tan(351.65711696)))))

Generation 523, Farm 139, MDD 42, Fitness 16.811479920861391:
(&G000000009219912341E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835288393E-08)))))
(97.89017641)+((t)*((49.99690709)+((t)/(Tan(351.65711696)))))

Generation 535, Farm 139, MDD 54, Fitness 16.8108983374592:
(&G000000009219972341E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835288393E-08)))))
(97.89077641)+((t)*((49.99690709)+((t)/(Tan(351.65711696)))))

Generation 540, Farm 139, MDD 59, Fitness 16.759837700952389:
(&G000000009229912361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835286393E-08)))))
(97.70982338)+((t)*((49.99690709)+((t)/(Tan(351.65713696)))))

Generation 546, Farm 139, MDD 65, Fitness 16.759825639478979:
(&G000000009229912361E-08)+((t)*((&G000000004900399899E-08)+((t)/(Tan(&G000000034835286393E-08)))))
(97.70982338)+((t)*((49.9969089)+((t)/(Tan(351.65713696)))))

Generation 548, Farm 139, MDD 67, Fitness 16.58351761814281:
(&G000000009129912361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835288393E-08)))))
(98.29017661)+((t)*((49.99690709)+((t)/(Tan(351.65711696)))))

Generation 552, Farm 139, MDD 71, Fitness 16.533117649350224:
(&G000000009219912341E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835284693E-08)))))
(97.89017641)+((t)*((49.99690709)+((t)/(Tan(351.65715303)))))

Generation 556, Farm 139, MDD 75, Fitness 16.36707610636017:
(&G000000009209912361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835286363E-08)))))
(97.90982338000001)+((t)*((49.99690709)+((t)/(Tan(351.65713663)))))

Generation 557, Farm 139, MDD 76, Fitness 16.364807391933962:
(&G000000009209912361E-08)+((t)*((&G000000004900393799E-08)+((t)/(Tan(&G000000034835286393E-08)))))
(97.90982338000001)+((t)*((49.99696709)+((t)/(Tan(351.65713696)))))

Generation 559, Farm 139, MDD 78, Fitness 16.364807374202556:
(&G000000009209912360E-08)+((t)*((&G000000004900393799E-08)+((t)/(Tan(&G000000034835286393E-08)))))
(97.90982339)+((t)*((49.99696709)+((t)/(Tan(351.65713696)))))

Generation 568, Farm 51, MDD 87, Fitness 16.009734944675351:
((t)*((-((&G000000000499300040E-08)*(t))+(&G000000004900020010E-08))))+(&G000000009010440082E-08)
((t)*((-((4.90399959)*(t))+(49.9997998))))+(99.80440082)

Generation 569, Farm 139, MDD 88, Fitness 15.002046482789453:
(&G000000009019912345E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835283393E-08)))))
(99.89017645)+((t)*((49.99690709)+((t)/(Tan(351.65716303)))))

Generation 578, Farm 139, MDD 97, Fitness 15.001480729102497:
(&G000000009019912345E-08)+((t)*((&G000000004900390799E-08)+((t)/(Tan(&G000000034835283393E-08)))))
(99.89017645)+((t)*((49.9969929)+((t)/(Tan(351.65716303)))))

Generation 583, Farm 139, MDD 98, Fitness 14.414561707583266:
(&G000000009029912361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.70982338)+((t)*((49.99690709)+((t)/(Tan(351.65719696)))))

Generation 590, Farm 139, MDD 91, Fitness 14.414557496670866:
(&G000000009029912961E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.70982938)+((t)*((49.99690709)+((t)/(Tan(351.65719696)))))

Generation 593, Farm 139, MDD 88, Fitness 14.414557430017769:
(&G000000009029912961E-08)+((t)*((&G000000004900399789E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.70982938)+((t)*((49.9969071)+((t)/(Tan(351.65719696)))))

Generation 594, Farm 139, MDD 87, Fitness 14.414390236334805:
(&G000000009029912361E-08)+((t)*((&G000000004900396799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.70982338)+((t)*((49.9969329)+((t)/(Tan(351.65719696)))))

Generation 597, Farm 139, MDD 84, Fitness 14.320514685280131:
(&G000000009019912961E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.89017061)+((t)*((49.99690709)+((t)/(Tan(351.65719696)))))

Generation 599, Farm 139, MDD 82, Fitness 14.314194951035969:
(&G000000009009912961E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.90982938000001)+((t)*((49.99690709)+((t)/(Tan(351.65719696)))))

Generation 601, Farm 139, MDD 80, Fitness 14.31419072585372:
(&G000000009009914361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.90984338)+((t)*((49.99690709)+((t)/(Tan(351.65719696)))))

Generation 607, Farm 139, MDD 74, Fitness 14.299619555115136:
(&G000000009002910361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.97019661)+((t)*((49.99690709)+((t)/(Tan(351.65719696)))))

Generation 619, Farm 139, MDD 62, Fitness 14.299510050424205:
(&G000000009002980361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280393E-08)))))
(99.97080338000001)+((t)*((49.99690709)+((t)/(Tan(351.65719696)))))

Generation 626, Farm 139, MDD 55, Fitness 14.297749168522362:
(&G000000009002910361E-08)+((t)*((&G000000004900399799E-08)+((t)/(Tan(&G000000034835280793E-08)))))
(99.97019661)+((t)*((49.99690709)+((t)/(Tan(351.65719296)))))

Generation 629, Farm 125, MDD 52, Fitness 14.128073457580387:
(&G000000009013150201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001920400240E-08)))
(99.83859798)-(((t)*(4.90299567))*((t)-(10.2040024)))

Generation 639, Farm 125, MDD 42, Fitness 14.127921587901012:
(&G000000009013150201E-08)-(((t)*(&G000000000499290334E-08))*((t)-(&G000000001920400240E-08)))
(99.83859798)-(((t)*(4.90299635))*((t)-(10.2040024)))

Generation 641, Farm 125, MDD 40, Fitness 14.127630379629608:
(&G000000009013150201E-08)-(((t)*(&G000000000499290237E-08))*((t)-(&G000000001920400240E-08)))
(99.83859798)-(((t)*(4.90299767))*((t)-(10.2040024)))

Generation 642, Farm 125, MDD 39, Fitness 13.982625756495239:
(&G000000009082550201E-08)-(((t)*(&G000000000499290037E-08))*((t)-(&G000000001920100440E-08)))
(99.17459798)-(((t)*(4.90299967))*((t)-(10.20199559)))

Generation 646, Farm 125, MDD 35, Fitness 13.891181516369441:
(&G000000009082550201E-08)-(((t)*(&G000000000499280037E-08))*((t)-(&G000000001920200240E-08)))
(99.17459798)-(((t)*(4.90280032))*((t)-(10.2020024)))

Generation 647, Farm 125, MDD 34, Fitness 13.817909849369116:
(&G000000009063550401E-08)-(((t)*(&G000000000499290037E-08))*((t)-(&G000000001920100240E-08)))
(99.36540401000001)-(((t)*(4.90299967))*((t)-(10.20199759)))

Generation 653, Farm 125, MDD 32, Fitness 13.487945008498752:
(&G000000009013150201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910400240E-08)))
(99.83859798)-(((t)*(4.90299567))*((t)-(10.19599759)))

Generation 666, Farm 125, MDD 45, Fitness 13.487944952703909:
(&G000000009013150221E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910400240E-08)))
(99.83859778)-(((t)*(4.90299567))*((t)-(10.19599759)))

Generation 668, Farm 125, MDD 47, Fitness 13.487934975060032:
(&G000000009013153201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910400240E-08)))
(99.83856201)-(((t)*(4.90299567))*((t)-(10.19599759)))

Generation 671, Farm 125, MDD 50, Fitness 13.487551083483108:
(&G000000009013150201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910300240E-08)))
(99.83859798)-(((t)*(4.90299567))*((t)-(10.1960024)))

Generation 672, Farm 125, MDD 51, Fitness 13.483443463368417:
(&G000000009012150201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910400240E-08)))
(99.82140201)-(((t)*(4.90299567))*((t)-(10.19599759)))

Generation 675, Farm 125, MDD 54, Fitness 13.47214999938128:
(&G000000009033150201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910400240E-08)))
(99.63859798)-(((t)*(4.90299567))*((t)-(10.19599759)))

Generation 689, Farm 125, MDD 68, Fitness 13.404884074952891:
(&G000000009013150201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910100240E-08)))
(99.83859798)-(((t)*(4.90299567))*((t)-(10.1980024)))

Generation 694, Farm 125, MDD 73, Fitness 13.401987604142269:
(&G000000009013150201E-08)-(((t)*(&G000000000499290037E-08))*((t)-(&G000000001910100240E-08)))
(99.83859798)-(((t)*(4.90299967))*((t)-(10.1980024)))

Generation 696, Farm 125, MDD 75, Fitness 13.400490238916928:
(&G000000009012150201E-08)-(((t)*(&G000000000499290436E-08))*((t)-(&G000000001910109240E-08)))
(99.82140201)-(((t)*(4.90299566))*((t)-(10.19809759)))

Generation 698, Farm 125, MDD 77, Fitness 13.389166187784868:
(&G000000009033100201E-08)-(((t)*(&G000000000499290437E-08))*((t)-(&G000000001910200240E-08)))
(99.63800200999999)-(((t)*(4.90299567))*((t)-(10.19799759)))

Generation 699, Farm 125, MDD 78, Fitness 13.385880290774345:
(&G000000009037150201E-08)-(((t)*(&G000000000499290430E-08))*((t)-(&G000000001910100240E-08)))
(99.67859798000001)-(((t)*(4.9029956))*((t)-(10.1980024)))

Generation 706, Farm 125, MDD 85, Fitness 13.38405690996192:
(&G000000009037150201E-08)-(((t)*(&G000000000499290030E-08))*((t)-(&G000000001910101240E-08)))
(99.67859798000001)-(((t)*(4.9029996))*((t)-(10.19801759)))

Generation 709, Farm 125, MDD 88, Fitness 13.381938431036614:
(&G000000009037150201E-08)-(((t)*(&G000000000499390430E-08))*((t)-(&G000000001910100240E-08)))
(99.67859798000001)-(((t)*(4.90300439))*((t)-(10.1980024)))

Generation 714, Farm 125, MDD 93, Fitness 13.379701405063146:
(&G000000009037150201E-08)-(((t)*(&G000000000499390930E-08))*((t)-(&G000000001910100240E-08)))
(99.67859798000001)-(((t)*(4.9030096))*((t)-(10.1980024)))

Generation 721, Farm 125, MDD 100, Fitness 13.379650212667809:
(&G000000009037050201E-08)-(((t)*(&G000000000499390930E-08))*((t)-(&G000000001910100240E-08)))
(99.67940201)-(((t)*(4.9030096))*((t)-(10.1980024)))

Generation 726, Farm 125, MDD 95, Fitness 13.354479846043136:
(&G000000009037150201E-08)-(((t)*(&G000000000499399930E-08))*((t)-(&G000000001910100240E-08)))
(99.67859798000001)-(((t)*(4.90309039))*((t)-(10.1980024)))

```
Generation 735, Farm 125, MDD 86, Fitness 13.347895909396211:
(&G000000009037150201E-08)-(((t)*(&G000000000499380930E-08))*((t)-(&G000000001910100240E-08)))
(99.67859798000001)-(((t)*(4.90319039))*((t)-(10.1980024)))

Generation 741, Farm 125, MDD 80, Fitness 13.34784112627009:
(&G000000009037150201E-08)-(((t)*(&G000000000499381930E-08))*((t)-(&G000000001910100240E-08)))
(99.67859798000001)-(((t)*(4.9031896))*((t)-(10.1980024)))

Generation 744, Farm 125, MDD 77, Fitness 13.347597100199902:
(&G000000009037050901E-08)-(((t)*(&G000000000499380930E-08))*((t)-(&G000000001910100240E-08)))
(99.67940998)-(((t)*(4.90319039))*((t)-(10.1980024)))

Generation 748, Farm 125, MDD 73, Fitness 13.347593820897721:
(&G000000009037050201E-08)-(((t)*(&G000000000499380930E-08))*((t)-(&G000000001910100200E-08)))
(99.67940201)-(((t)*(4.90319039))*((t)-(10.198002)))

Generation 749, Farm 125, MDD 72, Fitness 13.34756613601453:
(&G000000009037050901E-08)-(((t)*(&G000000000499380930E-08))*((t)-(&G000000001910100040E-08)))
(99.67940998)-(((t)*(4.90319039))*((t)-(10.1980004)))

Generation 753, Farm 125, MDD 68, Fitness 13.347561320602772:
(&G000000009037050901E-08)-(((t)*(&G000000000499380937E-08))*((t)-(&G000000001910100040E-08)))
(99.67940998)-(((t)*(4.90319032))*((t)-(10.1980004)))

Generation 766, Farm 125, MDD 55, Fitness 13.347558781807908:
(&G000000009037053901E-08)-(((t)*(&G000000000499380930E-08))*((t)-(&G000000001910100040E-08)))
(99.67943001)-(((t)*(4.90319039))*((t)-(10.1980004)))

Generation 767, Farm 125, MDD 54, Fitness 13.314130211798592:
(&G000000009017050901E-08)-(((t)*(&G000000000499380930E-08))*((t)-(&G000000001910100040E-08)))
(99.87940998000001)-(((t)*(4.90319039))*((t)-(10.1980004)))

Generation 777, Farm 125, MDD 44, Fitness 13.312092406828871:
(&G000000009017050901E-08)-(((t)*(&G000000000499380137E-08))*((t)-(&G000000001910100040E-08)))
(99.87940998000001)-(((t)*(4.90319832))*((t)-(10.1980004)))

Generation 790, Farm 125, MDD 31, Fitness 13.312075181703593:
(&G000000009017050901E-08)-(((t)*(&G000000000499380130E-08))*((t)-(&G000000001910100040E-08)))
(99.87940998000001)-(((t)*(4.90319839))*((t)-(10.1980004)))

Generation 793, Farm 125, MDD 32, Fitness 13.312055512197233:
(&G000000009017050901E-08)-(((t)*(&G000000000499380147E-08))*((t)-(&G000000001910100040E-08)))
(99.87940998000001)-(((t)*(4.90319847))*((t)-(10.1980004)))

Generation 796, Farm 125, MDD 35, Fitness 13.309489355428513:
(&G000000009017010901E-08)-(((t)*(&G000000000499370930E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.9032096))*((t)-(10.1980004)))

Generation 801, Farm 125, MDD 40, Fitness 13.309489002010247:
(&G000000009017050901E-08)-(((t)*(&G000000000499370930E-08))*((t)-(&G000000001910100040E-08)))
(99.87940998000001)-(((t)*(4.9032096))*((t)-(10.1980004)))

Generation 805, Farm 125, MDD 44, Fitness 13.309472076042992:
(&G000000009017010901E-08)-(((t)*(&G000000000499370938E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.90320968))*((t)-(10.1980004)))

Generation 815, Farm 125, MDD 54, Fitness 13.309448345366482:
(&G000000009017010901E-08)-(((t)*(&G000000000499370920E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.90320979))*((t)-(10.1980004)))

Generation 817, Farm 125, MDD 56, Fitness 13.309446189667398:
(&G000000009017010901E-08)-(((t)*(&G000000000499370910E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.9032098))*((t)-(10.1980004)))

Generation 822, Farm 125, MDD 61, Fitness 13.305585564093427:
(&G000000009017010901E-08)-(((t)*(&G000000000499373930E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.90323039))*((t)-(10.1980004)))

Generation 825, Farm 125, MDD 64, Fitness 13.305520421202106:
(&G000000009017010901E-08)-(((t)*(&G000000000499373980E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.9032308))*((t)-(10.1980004)))

Generation 826, Farm 125, MDD 65, Fitness 13.30092435382968:
(&G000000009017010901E-08)-(((t)*(&G000000000499379930E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.90329039))*((t)-(10.1980004)))

Generation 831, Farm 125, MDD 70, Fitness 13.300922150954953:
(&G000000009017010901E-08)-(((t)*(&G000000000499378938E-08))*((t)-(&G000000001910100040E-08)))
(99.87980998)-(((t)*(4.90328968))*((t)-(10.1980004)))

Generation 838, Farm 125, MDD 77, Fitness 13.300900133867312:
(&G000000009017000901E-08)-(((t)*(&G000000000499379930E-08))*((t)-(&G000000001910100040E-08)))
(99.87999001)-(((t)*(4.90329039))*((t)-(10.1980004)))
```

Generation 840, Farm 125, MDD 79, Fitness 13.300898142163701:
(&G000000009017000904E-08)-(((t)*(&G000000000499378938E-08))*((t)-(&G000000001910100040E-08)))
(99.87999004)-(((t)*(4.90328968))*((t)-(10.1980004)))

Generation 845, Farm 125, MDD 84, Fitness 13.297115187511864:
(&G000000009007010901E-08)-(((t)*(&G000000000499379930E-08))*((t)-(&G000000001910100040E-08)))
(99.92019001)-(((t)*(4.90329039))*((t)-(10.1980004)))

Generation 847, Farm 125, MDD 86, Fitness 13.297030409744311:
(&G000000009007110901E-08)-(((t)*(&G000000000499379930E-08))*((t)-(&G000000001910100040E-08)))
(99.92180998000001)-(((t)*(4.90329039))*((t)-(10.1980004)))

Generation 853, Farm 125, MDD 92, Fitness 13.296740258776365:
(&G000000009007900904E-08)-(((t)*(&G000000000499378938E-08))*((t)-(&G000000001910100040E-08)))
(99.92999004000001)-(((t)*(4.90328968))*((t)-(10.1980004)))

Generation 855, Farm 125, MDD 94, Fitness 13.296119281576976:
(&G000000009007990901E-08)-(((t)*(&G000000000499379061E-08))*((t)-(&G000000001910100040E-08)))
(99.92900998)-(((t)*(4.90329938))*((t)-(10.1980004)))

Generation 858, Farm 125, MDD 97, Fitness 13.293897700353597:
(&G000000009017010901E-08)-(((t)*(&G000000000499379938E-08))*((t)-(&G000000001910260040E-08)))
(99.87980998)-(((t)*(4.90329031))*((t)-(10.19739959)))

Generation 867, Farm 125, MDD 94, Fitness 13.290093971879813:
(&G000000009007010901E-08)-(((t)*(&G000000000499379938E-08))*((t)-(&G000000001910260040E-08)))
(99.92019001)-(((t)*(4.90329031))*((t)-(10.19739959)))

Generation 877, Farm 125, MDD 84, Fitness 13.289447009726199:
(&G000000009007010901E-08)-(((t)*(&G000000000499379330E-08))*((t)-(&G000000001910260040E-08)))
(99.92019001)-(((t)*(4.90329639))*((t)-(10.19739959)))

Generation 882, Farm 125, MDD 79, Fitness 13.289024291852709:
(&G000000009007100901E-08)-(((t)*(&G000000000499379030E-08))*((t)-(&G000000001910260040E-08)))
(99.92199001)-(((t)*(4.9032996))*((t)-(10.19739959)))

Generation 883, Farm 125, MDD 78, Fitness 13.288929882336607:
(&G000000009007910901E-08)-(((t)*(&G000000000499379330E-08))*((t)-(&G000000001910261040E-08)))
(99.92980998)-(((t)*(4.90329639))*((t)-(10.1973804)))

Generation 885, Farm 125, MDD 76, Fitness 13.288929875060052:
(&G000000009007910901E-08)-(((t)*(&G000000000499379330E-08))*((t)-(&G000000001910261041E-08)))
(99.92980998)-(((t)*(4.90329639))*((t)-(10.19738041)))

Generation 887, Farm 125, MDD 74, Fitness 13.288921847071233:
(&G000000009007900901E-08)-(((t)*(&G000000000499379330E-08))*((t)-(&G000000001910261040E-08)))
(99.92999001)-(((t)*(4.90329639))*((t)-(10.1973804)))

Generation 888, Farm 125, MDD 73, Fitness 13.288878847908968:
(&G000000009007414901E-08)-(((t)*(&G000000000499379038E-08))*((t)-(&G000000001910260090E-08)))
(99.92415001000001)-(((t)*(4.90329968))*((t)-(10.197399)))

Generation 905, Farm 125, MDD 56, Fitness 13.288382996974804:
(&G000000009007010901E-08)-(((t)*(&G000000000499369939E-08))*((t)-(&G000000001910260040E-08)))
(99.92019001)-(((t)*(4.90330969))*((t)-(10.19739959)))

Generation 910, Farm 125, MDD 51, Fitness 13.288311598349624:
(&G000000009007090901E-08)-(((t)*(&G000000000499369938E-08))*((t)-(&G000000001910260340E-08)))
(99.92099001)-(((t)*(4.90330968))*((t)-(10.1973964)))

Generation 912, Farm 125, MDD 49, Fitness 13.287928765220885:
(&G000000009004414901E-08)-(((t)*(&G000000000499379038E-08))*((t)-(&G000000001910260090E-08)))
(99.95584998)-(((t)*(4.90329968))*((t)-(10.197399)))

Generation 920, Farm 125, MDD 41, Fitness 13.287928602378857:
(&G000000009004424901E-08)-(((t)*(&G000000000499379038E-08))*((t)-(&G000000001910260090E-08)))
(99.95575001)-(((t)*(4.90329968))*((t)-(10.197399)))

Generation 922, Farm 125, MDD 39, Fitness 13.287927645741776:
(&G000000009004414901E-08)-(((t)*(&G000000000499379038E-08))*((t)-(&G000000001910260890E-08)))
(99.95584998)-(((t)*(4.90329968))*((t)-(10.197391)))

Generation 926, Farm 125, MDD 35, Fitness 13.286525347864922:
(&G000000009003010901E-08)-(((t)*(&G000000000499369939E-08))*((t)-(&G000000001910260040E-08)))
(99.96019001000001)-(((t)*(4.90330969))*((t)-(10.19739959)))

Generation 940, Farm 125, MDD 39, Fitness 13.28652405588473:
(&G000000009003010901E-08)-(((t)*(&G000000000499369929E-08))*((t)-(&G000000001910260040E-08)))
(99.96019001000001)-(((t)*(4.9033097))*((t)-(10.19739959)))

Generation 941, Farm 125, MDD 40, Fitness 13.286515134996508:
(&G000000009003310901E-08)-(((t)*(&G000000000499369939E-08))*((t)-(&G000000001910260040E-08)))
(99.96380997999999)-(((t)*(4.90330969))*((t)-(10.19739959)))

```
Generation 946, Farm 125, MDD 45, Fitness 13.286510381574123:
(&G000000009003010901E-08)-(((t)*(&G000000000499369939E-08))*((t)-(&G000000001910261040E-08)))
(99.96019001000001)-(((t)*(4.90330969))*((t)-(10.1973804)))

Generation 947, Farm 125, MDD 46, Fitness 13.286447035788797:
(&G000000009003010901E-08)-(((t)*(&G000000000499368939E-08))*((t)-(&G000000001910260040E-08)))
(99.96019001000001)-(((t)*(4.9033103))*((t)-(10.19739959)))

Generation 951, Farm 125, MDD 50, Fitness 13.283678028331943:
(&G000000009003010901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910260040E-08)))
(99.96019001000001)-(((t)*(4.90336969))*((t)-(10.19739959)))

Generation 959, Farm 125, MDD 58, Fitness 13.283267559064516:
(&G000000009003310901E-08)-(((t)*(&G000000000499364939E-08))*((t)-(&G000000001910260040E-08)))
(99.96380997999999)-(((t)*(4.9033503))*((t)-(10.19739959)))

Generation 962, Farm 125, MDD 61, Fitness 13.283264263420975:
(&G000000009003410901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910260040E-08)))
(99.96419001)-(((t)*(4.90336969))*((t)-(10.19739959)))

Generation 967, Farm 125, MDD 66, Fitness 13.28316880610741:
(&G000000009003310901E-08)-(((t)*(&G000000000499364339E-08))*((t)-(&G000000001910260040E-08)))
(99.96380997999999)-(((t)*(4.9033563))*((t)-(10.19739959)))

Generation 973, Farm 125, MDD 72, Fitness 13.281675438866035:
(&G000000009001410901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910260040E-08)))
(99.98419001000001)-(((t)*(4.90336969))*((t)-(10.19739959)))

Generation 975, Farm 125, MDD 74, Fitness 13.280990952134665:
(&G000000009000010901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910260040E-08)))
(99.99980997999999)-(((t)*(4.90336969))*((t)-(10.19739959)))

Generation 978, Farm 125, MDD 77, Fitness 13.280971699658576:
(&G000000009000010901E-08)-(((t)*(&G000000000499362939E-08))*((t)-(&G000000001910260040E-08)))
(99.99980997999999)-(((t)*(4.9033703))*((t)-(10.19739959)))

Generation 981, Farm 125, MDD 80, Fitness 13.280874678582824:
(&G000000009000010901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910262040E-08)))
(99.99980997999999)-(((t)*(4.90336969))*((t)-(10.19737959)))

Generation 985, Farm 125, MDD 84, Fitness 13.280556720556236:
(&G000000009000010901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910280040E-08)))
(99.99980997999999)-(((t)*(4.90336969))*((t)-(10.19719959)))

Generation 990, Farm 125, MDD 89, Fitness 13.28055559410474:
(&G000000009000014901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910280040E-08)))
(99.99984997999999)-(((t)*(4.90336969))*((t)-(10.19719959)))

Generation 997, Farm 125, MDD 96, Fitness 13.28054890080598:
(&G000000009000010901E-08)-(((t)*(&G000000000499362939E-08))*((t)-(&G000000001910269040E-08)))
(99.99980997999999)-(((t)*(4.9033703))*((t)-(10.1973004)))

Generation 998, Farm 125, MDD 97, Fitness 13.280547246892775:
(&G000000019000014901E-08)-(((t)*(&G000000000499363939E-08))*((t)-(&G000000001910280040E-08)))
(100.00015001)-(((t)*(4.90336969))*((t)-(10.19719959)))

Generation 1000, Farm 125, MDD 99, Fitness 13.280432310803048:
(&G000000009000010901E-08)-(((t)*(&G000000000499362639E-08))*((t)-(&G000000001910269049E-08)))
(99.99980997999999)-(((t)*(4.90337369))*((t)-(10.19730049)))

Generation 1005, Farm 125, MDD 96, Fitness 13.280426398701543:
(&G000000009000010901E-08)-(((t)*(&G000000000499362619E-08))*((t)-(&G000000001910269049E-08)))
(99.99980997999999)-(((t)*(4.90337389))*((t)-(10.19730049)))

Generation 1007, Farm 125, MDD 94, Fitness 13.280425825114602:
(&G000000009000000901E-08)-(((t)*(&G000000000499362639E-08))*((t)-(&G000000001910269040E-08)))
(99.99999001)-(((t)*(4.90337369))*((t)-(10.1973004)))

Generation 1010, Farm 125, MDD 91, Fitness 13.280118444937942:
(&G000000009000014901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280040E-08)))
(99.99984997999999)-(((t)*(4.9033903))*((t)-(10.19719959)))

Generation 1012, Farm 125, MDD 89, Fitness 13.280118443526082:
(&G000000009000014901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280030E-08)))
(99.99984997999999)-(((t)*(4.9033903))*((t)-(10.1971996)))

Generation 1014, Farm 125, MDD 87, Fitness 13.280117232741254:
(&G000000009000010901E-08)-(((t)*(&G000000000499361939E-08))*((t)-(&G000000001910280040E-08)))
(99.99980997999999)-(((t)*(4.90338969))*((t)-(10.19719959)))

Generation 1024, Farm 125, MDD 77, Fitness 13.280109668591628:
(&G000000009000000901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280040E-08)))
(99.99999001)-(((t)*(4.9033903))*((t)-(10.19719959)))
```

Generation 1028, Farm 125, MDD 73, Fitness 13.280109639792217:
(&G000000009000000901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280010E-08)))
(99.99999001)-(((t)*(4.9033903))*((t)-(10.1971998)))

Generation 1030, Farm 125, MDD 71, Fitness 13.280093479022854:
(&G000000019000024901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280040E-08)))
(100.00024998)-(((t)*(4.9033903))*((t)-(10.19719959)))

Generation 1051, Farm 125, MDD 50, Fitness 13.280092237267857:
(&G000000019000027901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280040E-08)))
(100.00027001)-(((t)*(4.9033903))*((t)-(10.19719959)))

Generation 1053, Farm 125, MDD 48, Fitness 13.279594587248818:
(&G000000019000924901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280040E-08)))
(100.00975001)-(((t)*(4.9033903))*((t)-(10.19719959)))

Generation 1058, Farm 125, MDD 43, Fitness 13.279212887513008:
(&G000000019004024901E-08)-(((t)*(&G000000000499360939E-08))*((t)-(&G000000001910280040E-08)))
(100.04024998)-(((t)*(4.9033903))*((t)-(10.19719959)))

Generation 1066, Farm 125, MDD 35, Fitness 13.279176697661885:
(&G000000019004024901E-08)-(((t)*(&G000000000499360999E-08))*((t)-(&G000000001910280040E-08)))
(100.04024998)-(((t)*(4.9033909))*((t)-(10.19719959)))

Generation 1068, Farm 125, MDD 33, Fitness 13.278895137301063:
(&G000000019004024901E-08)-(((t)*(&G000000000499360339E-08))*((t)-(&G000000001910280040E-08)))
(100.04024998)-(((t)*(4.9033963))*((t)-(10.19719959)))

Generation 1084, Farm 125, MDD 43, Fitness 13.278638607270274:
(&G000000019004024901E-08)-(((t)*(&G000000000499350339E-08))*((t)-(&G000000001910280040E-08)))
(100.04024998)-(((t)*(4.90340369))*((t)-(10.19719959)))

Generation 1085, Farm 125, MDD 44, Fitness 13.278539800361644:
(&G000000019004024904E-08)-(((t)*(&G000000000499350939E-08))*((t)-(&G000000001910280040E-08)))
(100.04024995)-(((t)*(4.90340969))*((t)-(10.19719959)))

Generation 1086, Farm 125, MDD 45, Fitness 13.278539507668032:
(&G000000019004026904E-08)-(((t)*(&G000000000499350939E-08))*((t)-(&G000000001910280040E-08)))
(100.04026995)-(((t)*(4.90340969))*((t)-(10.19719959)))

Generation 1102, Farm 125, MDD 61, Fitness 13.278531761248169:
(&G000000019004024901E-08)-(((t)*(&G000000000499351999E-08))*((t)-(&G000000001910280040E-08)))
(100.04024998)-(((t)*(4.9034109))*((t)-(10.19719959)))

Generation 1111, Farm 125, MDD 70, Fitness 13.278529836148373:
(&G000000019004026904E-08)-(((t)*(&G000000000499350939E-08))*((t)-(&G000000001910280740E-08)))
(100.04026995)-(((t)*(4.90340969))*((t)-(10.1971924)))

Generation 1112, Farm 125, MDD 71, Fitness 13.27848952277682:
(&G000000019004526904E-08)-(((t)*(&G000000000499350939E-08))*((t)-(&G000000001910280040E-08)))
(100.04573004)-(((t)*(4.90340969))*((t)-(10.19719959)))

Generation 1114, Farm 125, MDD 73, Fitness 13.278486140284642:
(&G000000019004724904E-08)-(((t)*(&G000000000499350939E-08))*((t)-(&G000000001910280040E-08)))
(100.04775004)-(((t)*(4.90340969))*((t)-(10.19719959)))

Generation 1121, Farm 125, MDD 80, Fitness 13.278463419377136:
(&G000000019004926004E-08)-(((t)*(&G000000000499351999E-08))*((t)-(&G000000001910280040E-08)))
(100.04973995)-(((t)*(4.9034109))*((t)-(10.19719959)))

Generation 1128, Farm 125, MDD 87, Fitness 13.278439154459166:
(&G000000019004924901E-08)-(((t)*(&G000000000499351999E-08))*((t)-(&G000000001910285140E-08)))
(100.04975001)-(((t)*(4.9034109))*((t)-(10.19714159)))

Generation 1130, Farm 125, MDD 89, Fitness 13.278426930651047:
(&G000000019004926004E-08)-(((t)*(&G000000000499351799E-08))*((t)-(&G000000001910280040E-08)))
(100.04973995)-(((t)*(4.9034129))*((t)-(10.19719959)))

Generation 1137, Farm 125, MDD 96, Fitness 13.278426755014831:
(&G000000019004906004E-08)-(((t)*(&G000000000499351799E-08))*((t)-(&G000000001910280040E-08)))
(100.04993995)-(((t)*(4.9034129))*((t)-(10.19719959)))

Generation 1140, Farm 125, MDD 99, Fitness 13.27836811352361:
(&G000000019004926004E-08)-(((t)*(&G000000000499351299E-08))*((t)-(&G000000001910280940E-08)))
(100.04973995)-(((t)*(4.90341709))*((t)-(10.1971904)))

Generation 1150, Farm 125, MDD 91, Fitness 13.278330830650232:
(&G000000019004926004E-08)-(((t)*(&G000000000499351299E-08))*((t)-(&G000000001910284940E-08)))
(100.04973995)-(((t)*(4.90341709))*((t)-(10.1971504)))

Generation 1155, Farm 125, MDD 86, Fitness 13.278324122820871:
(&G000000019004926004E-08)-(((t)*(&G000000000499351099E-08))*((t)-(&G000000001910284049E-08)))
(100.04973995)-(((t)*(4.90341909))*((t)-(10.1971595)))

Generation 1157, Farm 125, MDD 84, Fitness 13.278323477621338:
(&G000000019004926004E-08)-(((t)*(&G000000000499351099E-08))*((t)-(&G000000001910284149E-08)))
(100.04973995)-(((t)*(4.90341909))*((t)-(10.19715849)))

Generation 1160, Farm 125, MDD 81, Fitness 13.278319799437265:
(&G000000019004926004E-08)-(((t)*(&G000000000499351099E-08))*((t)-(&G000000001910284940E-08)))
(100.04973995)-(((t)*(4.90341909))*((t)-(10.1971504)))

Generation 1165, Farm 125, MDD 76, Fitness 13.278318433212162:
(&G000000019004926004E-08)-(((t)*(&G000000000499351009E-08))*((t)-(&G000000001910284940E-08)))
(100.04973995)-(((t)*(4.9034199))*((t)-(10.1971504)))

Generation 1179, Farm 125, MDD 62, Fitness 13.278317205113533:
(&G000000019004926004E-08)-(((t)*(&G000000000499351029E-08))*((t)-(&G000000001910285040E-08)))
(100.04973995)-(((t)*(4.9034197))*((t)-(10.1971404)))

Generation 1198, Farm 125, MDD 43, Fitness 13.278316889598017:
(&G000000019004926504E-08)-(((t)*(&G000000000499352099E-08))*((t)-(&G000000001910285040E-08)))
(100.04973404)-(((t)*(4.9034209))*((t)-(10.1971404)))

Generation 1200, Farm 125, MDD 41, Fitness 13.278316517304166:
(&G000000019004926004E-08)-(((t)*(&G000000000499352029E-08))*((t)-(&G000000001910285040E-08)))
(100.04973995)-(((t)*(4.90342029))*((t)-(10.1971404)))

Generation 1204, Farm 125, MDD 37, Fitness 13.278312014560742:
(&G000000019005926004E-08)-(((t)*(&G000000000499351009E-08))*((t)-(&G000000001910284940E-08)))
(100.05026004)-(((t)*(4.9034199))*((t)-(10.1971504)))

Generation 1213, Farm 125, MDD 32, Fitness 13.278311544364087:
(&G000000019005929004E-08)-(((t)*(&G000000000499351009E-08))*((t)-(&G000000001910284940E-08)))
(100.05029995)-(((t)*(4.9034199))*((t)-(10.1971504)))

Generation 1214, Farm 125, MDD 33, Fitness 13.278310961559079:
(&G000000019005926004E-08)-(((t)*(&G000000000499351029E-08))*((t)-(&G000000001910285040E-08)))
(100.05026004)-(((t)*(4.9034197))*((t)-(10.1971404)))

Generation 1220, Farm 125, MDD 39, Fitness 13.278277086554246:
(&G000000019005326004E-08)-(((t)*(&G000000000499351009E-08))*((t)-(&G000000001910284940E-08)))
(100.05626004)-(((t)*(4.9034199))*((t)-(10.1971504)))

Generation 1237, Farm 125, MDD 56, Fitness 13.278276626395385:
(&G000000019005326004E-08)-(((t)*(&G000000000499351005E-08))*((t)-(&G000000001910284940E-08)))
(100.05626004)-(((t)*(4.90341994))*((t)-(10.1971504)))

Generation 1247, Farm 125, MDD 66, Fitness 13.278276170596829:
(&G000000019005326004E-08)-(((t)*(&G000000000499351001E-08))*((t)-(&G000000001910284940E-08)))
(100.05626004)-(((t)*(4.90341998))*((t)-(10.1971504)))

Generation 1255, Farm 125, MDD 74, Fitness 13.278276057328492:
(&G000000019005326004E-08)-(((t)*(&G000000000499351000E-08))*((t)-(&G000000001910284940E-08)))
(100.05626004)-(((t)*(4.90341999))*((t)-(10.1971504)))

Generation 1258, Farm 125, MDD 77, Fitness 13.278275948455127:
(&G000000019005306004E-08)-(((t)*(&G000000000499351000E-08))*((t)-(&G000000001910284900E-08)))
(100.05606004)-(((t)*(4.90341999))*((t)-(10.19715)))

Generation 1260, Farm 125, MDD 79, Fitness 13.278274939633759:
(&G000000019005326004E-08)-(((t)*(&G000000000499352009E-08))*((t)-(&G000000001910284940E-08)))
(100.05626004)-(((t)*(4.90342009))*((t)-(10.1971504)))

Generation 1269, Farm 125, MDD 88, Fitness 13.278274861758337:
(&G000000019005326000E-08)-(((t)*(&G000000000499352009E-08))*((t)-(&G000000001910284910E-08)))
(100.05626)-(((t)*(4.90342009))*((t)-(10.19715019)))

Generation 1273, Farm 125, MDD 92, Fitness 13.278272599607027:
(&G000000019005326000E-08)-(((t)*(&G000000000499352039E-08))*((t)-(&G000000001910284910E-08)))
(100.05626)-(((t)*(4.9034203))*((t)-(10.19715019)))

Generation 1274, Farm 125, MDD 93, Fitness 13.278266890679584:
(&G000000019005326004E-08)-(((t)*(&G000000000499352099E-08))*((t)-(&G000000001910284940E-08)))
(100.05626004)-(((t)*(4.9034209))*((t)-(10.1971504)))

Generation 1275, Farm 125, MDD 94, Fitness 13.278266798541669:
(&G000000019005326000E-08)-(((t)*(&G000000000499352099E-08))*((t)-(&G000000001910284910E-08)))
(100.05626)-(((t)*(4.9034209))*((t)-(10.19715019)))

Generation 1276, Farm 125, MDD 95, Fitness 13.278255763102322:
(&G000000019005326004E-08)-(((t)*(&G000000000499352209E-08))*((t)-(&G000000001910287940E-08)))
(100.05626004)-(((t)*(4.90342209))*((t)-(10.19712959)))

Generation 1277, Farm 125, MDD 96, Fitness 13.278255665927887:
(&G000000019005026004E-08)-(((t)*(&G000000000499352209E-08))*((t)-(&G000000001910287940E-08)))
(100.05973995)-(((t)*(4.90342209))*((t)-(10.19712959)))

Generation 1283, Farm 125, MDD 98, Fitness 13.278255646819854:
(&G000000019005326004E-08)-(((t)*(&G000000000499352209E-08))*((t)-(&G000000001910287900E-08)))
(100.05626004)-(((t)*(4.90342209))*((t)-(10.19712999)))

Generation 1285, Farm 125, MDD 96, Fitness 13.278252447893763:
(&G000000019005326000E-08)-(((t)*(&G000000000499352402E-08))*((t)-(&G000000001910284910E-08)))
(100.05626)-(((t)*(4.90342402))*((t)-(10.19715019)))

Generation 1287, Farm 125, MDD 94, Fitness 13.278252430192932:
(&G000000019005326000E-08)-(((t)*(&G000000000499352409E-08))*((t)-(&G000000001910284910E-08)))
(100.05626)-(((t)*(4.90342409))*((t)-(10.19715019)))

Generation 1290, Farm 125, MDD 91, Fitness 13.278225051243751:
(&G000000019005026004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287900E-08)))
(100.05973995)-(((t)*(4.9034279))*((t)-(10.19712999)))

Generation 1295, Farm 125, MDD 86, Fitness 13.278224799154208:
(&G000000019005021004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287900E-08)))
(100.05978004)-(((t)*(4.9034279))*((t)-(10.19712999)))

Generation 1302, Farm 125, MDD 79, Fitness 13.278223587212924:
(&G000000019005021004E-08)-(((t)*(&G000000000499352799E-08))*((t)-(&G000000001910287900E-08)))
(100.05978004)-(((t)*(4.90342709))*((t)-(10.19712999)))

Generation 1306, Farm 125, MDD 75, Fitness 13.278222030513684:
(&G000000019006026004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287900E-08)))
(100.06026004)-(((t)*(4.9034279))*((t)-(10.19712999)))

Generation 1318, Farm 125, MDD 63, Fitness 13.27822202839865:
(&G000000019006026044E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287900E-08)))
(100.06026044)-(((t)*(4.9034279))*((t)-(10.19712999)))

Generation 1321, Farm 125, MDD 60, Fitness 13.278221631992828:
(&G000000019006021004E-08)-(((t)*(&G000000000499352799E-08))*((t)-(&G000000001910287900E-08)))
(100.06021995)-(((t)*(4.90342709))*((t)-(10.19712999)))

Generation 1323, Farm 125, MDD 58, Fitness 13.278215440162272:
(&G000000019006226004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287907E-08)))
(100.06226004)-(((t)*(4.9034279))*((t)-(10.19712992)))

Generation 1330, Farm 125, MDD 51, Fitness 13.278215336195055:
(&G000000019006226004E-08)-(((t)*(&G000000000499352702E-08))*((t)-(&G000000001910287907E-08)))
(100.06226004)-(((t)*(4.90342797))*((t)-(10.19712992)))

Generation 1336, Farm 125, MDD 45, Fitness 13.278215132814056:
(&G000000019006226004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287807E-08)))
(100.06226004)-(((t)*(4.9034279))*((t)-(10.19712807)))

Generation 1342, Farm 125, MDD 39, Fitness 13.278215083025826:
(&G000000019006266404E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287907E-08)))
(100.06266404)-(((t)*(4.9034279))*((t)-(10.19712992)))

Generation 1346, Farm 125, MDD 35, Fitness 13.278215052351264:
(&G000000019006276004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287907E-08)))
(100.06273995)-(((t)*(4.9034279))*((t)-(10.19712992)))

Generation 1352, Farm 125, MDD 31, Fitness 13.278214949661223:
(&G000000019006226004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287507E-08)))
(100.06226004)-(((t)*(4.9034279))*((t)-(10.19712592)))

Generation 1353, Farm 125, MDD 32, Fitness 13.278214807933271:
(&G000000019006226004E-08)-(((t)*(&G000000000499352702E-08))*((t)-(&G000000001910287307E-08)))
(100.06226004)-(((t)*(4.90342797))*((t)-(10.19712392)))

Generation 1357, Farm 125, MDD 36, Fitness 13.278214590114668:
(&G000000019006266404E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287307E-08)))
(100.06266404)-(((t)*(4.9034279))*((t)-(10.19712392)))

Generation 1365, Farm 125, MDD 44, Fitness 13.278214575816081:
(&G000000019006269404E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287307E-08)))
(100.06269595)-(((t)*(4.9034279))*((t)-(10.19712392)))

Generation 1372, Farm 125, MDD 51, Fitness 13.278214559440107:
(&G000000019006276004E-08)-(((t)*(&G000000000499352709E-08))*((t)-(&G000000001910287307E-08)))
(100.06273995)-(((t)*(4.9034279))*((t)-(10.19712392)))

Generation 1374, Farm 125, MDD 53, Fitness 13.27821450910927:
(&G000000019006266404E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287307E-08)))
(100.06266404)-(((t)*(4.9034299))*((t)-(10.19712392)))

Generation 1383, Farm 125, MDD 62, Fitness 13.278214455235986:
(&G000000019006266404E-08)-(((t)*(&G000000000499352704E-08))*((t)-(&G000000001910287307E-08)))
(100.06266404)-(((t)*(4.90342795))*((t)-(10.19712392)))

Generation 1385, Farm 125, MDD 64, Fitness 13.278214429077828:
(&G000000019006266404E-08)-(((t)*(&G000000000499352703E-08))*((t)-(&G000000001910287307E-08)))
(100.06266404)-(((t)*(4.90342796))*((t)-(10.19712392)))

Generation 1386, Farm 125, MDD 65, Fitness 13.278213537096899:
(&G000000019006296404E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287307E-08)))
(100.06293595)-(((t)*(4.9034299))*((t)-(10.19712392)))

Generation 1397, Farm 125, MDD 76, Fitness 13.278213537063866:
(&G000000019006296403E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287307E-08)))
(100.06293596)-(((t)*(4.9034299))*((t)-(10.19712392)))

Generation 1398, Farm 125, MDD 77, Fitness 13.278210816084449:
(&G000000019006466604E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287307E-08)))
(100.06466604)-(((t)*(4.9034299))*((t)-(10.19712392)))

Generation 1401, Farm 125, MDD 80, Fitness 13.278210816081305:
(&G000000019006466602E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287307E-08)))
(100.06466602)-(((t)*(4.9034299))*((t)-(10.19712392)))

Generation 1408, Farm 125, MDD 87, Fitness 13.278210740737153:
(&G000000019006466602E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287377E-08)))
(100.06466602)-(((t)*(4.9034299))*((t)-(10.19712327)))

Generation 1409, Farm 125, MDD 88, Fitness 13.278210685173292:
(&G000000019006466602E-08)-(((t)*(&G000000000499352900E-08))*((t)-(&G000000001910287377E-08)))
(100.06466602)-(((t)*(4.90342999))*((t)-(10.19712327)))

Generation 1415, Farm 125, MDD 94, Fitness 13.278210646565764:
(&G000000019006466602E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287207E-08)))
(100.06466602)-(((t)*(4.9034299))*((t)-(10.19712207)))

Generation 1418, Farm 125, MDD 97, Fitness 13.278210619128476:
(&G000000019006466602E-08)-(((t)*(&G000000000499352909E-08))*((t)-(&G000000001910287007E-08)))
(100.06466602)-(((t)*(4.9034299))*((t)-(10.19712007)))

Generation 1421, Farm 125, MDD 100, Fitness 13.278210540765876:
(&G000000019006466602E-08)-(((t)*(&G000000000499352900E-08))*((t)-(&G000000001910287000E-08)))
(100.06466602)-(((t)*(4.90342999))*((t)-(10.19712)))

Generation 1427, Farm 125, MDD 94, Fitness 13.278210535407412:
(&G000000019006466602E-08)-(((t)*(&G000000000499352900E-08))*((t)-(&G000000001910287077E-08)))
(100.06466602)-(((t)*(4.90342999))*((t)-(10.19712072)))

Generation 1432, Farm 125, MDD 89, Fitness 13.27821053540737:
(&G000000019006466601E-08)-(((t)*(&G000000000499352900E-08))*((t)-(&G000000001910287077E-08)))
(100.06466601)-(((t)*(4.90342999))*((t)-(10.19712072)))

Generation 1435, Farm 125, MDD 86, Fitness 13.2782105352589:
(&G000000019006466602E-08)-(((t)*(&G000000000499352900E-08))*((t)-(&G000000001910287030E-08)))
(100.06466602)-(((t)*(4.90342999))*((t)-(10.19712039)))

Generation 1443, Farm 125, MDD 78, Fitness 13.278210535101577:
(&G000000019006466602E-08)-(((t)*(&G000000000499352900E-08))*((t)-(&G000000001910287067E-08)))
(100.06466602)-(((t)*(4.90342999))*((t)-(10.19712067)))

Generation 1448, Farm 125, MDD 73, Fitness 13.278210476542132:
(&G000000019006466602E-08)-(((t)*(&G000000000499353909E-08))*((t)-(&G000000001910287007E-08)))
(100.06466602)-(((t)*(4.90343009))*((t)-(10.19712007)))

Generation 1465, Farm 125, MDD 56, Fitness 13.278210472780311:
(&G000000019006469602E-08)-(((t)*(&G000000000499353909E-08))*((t)-(&G000000001910287007E-08)))
(100.06469397)-(((t)*(4.90343009))*((t)-(10.19712007)))

Generation 1470, Farm 125, MDD 51, Fitness 13.278210432734019:
(&G000000019006466602E-08)-(((t)*(&G000000000499353929E-08))*((t)-(&G000000001910287007E-08)))
(100.06466602)-(((t)*(4.90343029))*((t)-(10.19712007)))

Generation 1487, Farm 125, MDD 34, Fitness 13.278210419562365:
(&G000000019006469602E-08)-(((t)*(&G000000000499353929E-08))*((t)-(&G000000001910287007E-08)))
(100.06469397)-(((t)*(4.90343029))*((t)-(10.19712007)))

Generation 1493, Farm 125, MDD 32, Fitness 13.278210372892465:
(&G000000019006486602E-08)-(((t)*(&G000000000499353929E-08))*((t)-(&G000000001910287007E-08)))
(100.06486602)-(((t)*(4.90343029))*((t)-(10.19712007)))

Generation 1495, Farm 125, MDD 34, Fitness 13.278210370768516:
(&G000000019006496602E-08)-(((t)*(&G000000000499353929E-08))*((t)-(&G000000001910287007E-08)))
(100.06493397)-(((t)*(4.90343029))*((t)-(10.19712007)))

Generation 1508, Farm 125, MDD 47, Fitness 13.278210370655207:
(&G000000019006496602E-08)-(((t)*(&G000000000499353929E-08))*((t)-(&G000000001910287005E-08)))
(100.06493397)-(((t)*(4.90343029))*((t)-(10.19712005)))

Generation 1519, Farm 125, MDD 58, Fitness 13.278210370539988:
(&G0000000190064976O2E-08)-(((t)*(&G000000000499353929E-08))*((t)-(&G00000000191028700 7E-08)))
(100.06492602)-(((t)*(4.90343029))*((t)-(10.19712007)))

Generation 1522, Farm 125, MDD 61, Fitness 13.278210370321347:
(&G0000000190064986O2E-08)-(((t)*(&G000000000499353929E-08))*((t)-(&G00000000191028700 5E-08)))
(100.06491397)-(((t)*(4.90343029))*((t)-(10.19712005)))

Generation 1523, Farm 125, MDD 62, Fitness 13.27821034736967:
(&G0000000190064966O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028700 7E-08)))
(100.06493397)-(((t)*(4.9034305))*((t)-(10.19712007)))

Generation 1534, Farm 125, MDD 73, Fitness 13.278210345504817:
(&G0000000190064960O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028700 7E-08)))
(100.06493997)-(((t)*(4.9034305))*((t)-(10.19712007)))

Generation 1542, Farm 125, MDD 81, Fitness 13.278210345308718:
(&G0000000190064966O2E-08)-(((t)*(&G000000000499353949E-08))*((t)-(&G00000000191028700 5E-08)))
(100.06493397)-(((t)*(4.90343049))*((t)-(10.19712005)))

Generation 1543, Farm 125, MDD 82, Fitness 13.278210331961198:
(&G0000000190064906O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028700 7E-08)))
(100.06499397)-(((t)*(4.9034305))*((t)-(10.19712007)))

Generation 1544, Farm 125, MDD 83, Fitness 13.278210331959214:
(&G0000000190064906O1E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028700 7E-08)))
(100.06499398)-(((t)*(4.9034305))*((t)-(10.19712007)))

Generation 1546, Farm 125, MDD 85, Fitness 13.278210322972379:
(&G0000000190065966O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028700 7E-08)))
(100.06506602)-(((t)*(4.9034305))*((t)-(10.19712007)))

Generation 1557, Farm 125, MDD 96, Fitness 13.278210321409293:
(&G0000000190065966O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028700 0E-08)))
(100.06506602)-(((t)*(4.9034305))*((t)-(10.19712)))

Generation 1569, Farm 125, MDD 92, Fitness 13.278210321378046:
(&G0000000190065966O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028700 0E-08)))
(100.06506662)-(((t)*(4.9034305))*((t)-(10.19712)))

Generation 1575, Farm 125, MDD 86, Fitness 13.278210319032617:
(&G0000000190065966O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028801 7E-08)))
(100.06506602)-(((t)*(4.9034305))*((t)-(10.19711987)))

Generation 1580, Farm 125, MDD 81, Fitness 13.278210316033592:
(&G0000000190065966O2E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06506602)-(((t)*(4.9034305))*((t)-(10.19711952)))

Generation 1589, Farm 125, MDD 72, Fitness 13.278210316023099:
(&G0000000190065966226E-08)-(((t)*(&G000000000499353959E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06506622)-(((t)*(4.9034305))*((t)-(10.19711952)))

Generation 1596, Farm 125, MDD 65, Fitness 13.278210315277009:
(&G0000000190065966226E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06506622)-(((t)*(4.90343056))*((t)-(10.19711952)))

Generation 1603, Farm 125, MDD 58, Fitness 13.278210314459596:
(&G0000000190065966O2E-08)-(((t)*(&G000000000499353955E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06506602)-(((t)*(4.90343054))*((t)-(10.19711952)))

Generation 1609, Farm 125, MDD 52, Fitness 13.278210314423738:
(&G0000000190065966226E-08)-(((t)*(&G000000000499353956E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06506622)-(((t)*(4.90343053))*((t)-(10.19711952)))

Generation 1611, Farm 125, MDD 50, Fitness 13.278210312610863:
(&G0000000190065986226E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06508622)-(((t)*(4.90343056))*((t)-(10.19711952)))

Generation 1619, Farm 125, MDD 42, Fitness 13.278210312602937:
(&G0000000190065986296E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06508629)-(((t)*(4.90343056))*((t)-(10.19711952)))

Generation 1626, Farm 125, MDD 35, Fitness 13.278210311812408:
(&G0000000190065996226E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G00000000191028804 7E-08)))
(100.06509377)-(((t)*(4.90343056))*((t)-(10.19711952)))

Generation 1644, Farm 125, MDD 43, Fitness 13.27821031131476:
(&G0000000190065996226E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G00000000191028806 7E-08)))
(100.06509377)-(((t)*(4.90343056))*((t)-(10.19711932)))

Generation 1655, Farm 125, MDD 54, Fitness 13.278210311287385:
(&G0000000190065996226E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G00000000191028806 2E-08)))
(100.06509377)-(((t)*(4.90343056))*((t)-(10.19711937)))

Generation 1658, Farm 125, MDD 57, Fitness 13.278210310897517:
(&G000000019006599122E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G000000001910288067E-08)))
(100.06509822)-(((t)*(4.90343056))*((t)-(10.19711932)))

Generation 1666, Farm 125, MDD 65, Fitness 13.278210310744717:
(&G000000019006589622E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G000000001910288047E-08)))
(100.06510622)-(((t)*(4.90343056))*((t)-(10.19711952)))

Generation 1675, Farm 125, MDD 74, Fitness 13.278210310247069:
(&G000000019006589622E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G000000001910288067E-08)))
(100.06510622)-(((t)*(4.90343056))*((t)-(10.19711932)))

Generation 1682, Farm 125, MDD 81, Fitness 13.278210309946243:
(&G000000019006583622E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G000000001910288047E-08)))
(100.06516622)-(((t)*(4.90343056))*((t)-(10.19711952)))

Generation 1688, Farm 125, MDD 87, Fitness 13.278210308925225:
(&G000000019006586122E-08)-(((t)*(&G000000000499353953E-08))*((t)-(&G000000001910288067E-08)))
(100.06513822)-(((t)*(4.90343056))*((t)-(10.19711932)))

Generation 1694, Farm 125, MDD 93, Fitness 13.278210308856352:
(&G000000019006580622E-08)-(((t)*(&G000000000499353950E-08))*((t)-(&G000000001910288047E-08)))
(100.06519377)-(((t)*(4.90343059))*((t)-(10.19711952)))

Generation 1697, Farm 125, MDD 96, Fitness 13.278210308836019:
(&G000000019006580662E-08)-(((t)*(&G000000000499353950E-08))*((t)-(&G000000001910288047E-08)))
(100.06519337)-(((t)*(4.90343059))*((t)-(10.19711952)))

Generation 1700, Farm 125, MDD 99, Fitness 13.278210307401974:
(&G000000019006584622E-08)-(((t)*(&G000000000499353950E-08))*((t)-(&G000000001910288067E-08)))
(100.06515377)-(((t)*(4.90343059))*((t)-(10.19711932)))

Generation 1712, Farm 125, MDD 89, Fitness 13.27821030726572:
(&G000000019006584022E-08)-(((t)*(&G000000000499353950E-08))*((t)-(&G000000001910288067E-08)))
(100.06515977)-(((t)*(4.90343059))*((t)-(10.19711932)))

Generation 1716, Farm 125, MDD 85, Fitness 13.278210307199558:
(&G000000019006583622E-08)-(((t)*(&G000000000499353950E-08))*((t)-(&G000000001910288067E-08)))
(100.06516622)-(((t)*(4.90343059))*((t)-(10.19711932)))

Generation 1727, Farm 125, MDD 74, Fitness 13.278210307169696:
(&G000000019006584022E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288067E-08)))
(100.06515977)-(((t)*(4.9034306))*((t)-(10.19711932)))

Generation 1751, Farm 125, MDD 50, Fitness 13.278210307168024:
(&G000000019006584012E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288067E-08)))
(100.06515982)-(((t)*(4.9034306))*((t)-(10.19711932)))

Generation 1756, Farm 125, MDD 45, Fitness 13.278210307141638:
(&G000000019006584022E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288069E-08)))
(100.06515977)-(((t)*(4.9034306))*((t)-(10.1971193)))

Generation 1774, Farm 125, MDD 33, Fitness 13.278210307139966:
(&G000000019006584012E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288069E-08)))
(100.06515982)-(((t)*(4.9034306))*((t)-(10.1971193)))

Generation 1776, Farm 125, MDD 35, Fitness 13.278210307133653:
(&G000000019006584022E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288079E-08)))
(100.06515977)-(((t)*(4.9034306))*((t)-(10.19711929)))

Generation 1778, Farm 125, MDD 37, Fitness 13.278210306898893:
(&G000000019006582022E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288067E-08)))
(100.06517977)-(((t)*(4.9034306))*((t)-(10.19711932)))

Generation 1783, Farm 125, MDD 42, Fitness 13.278210306873909:
(&G000000019006581022E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288069E-08)))
(100.06518022)-(((t)*(4.9034306))*((t)-(10.1971193)))

Generation 1785, Farm 125, MDD 44, Fitness 13.278210306870836:
(&G000000019006582022E-08)-(((t)*(&G000000000499353960E-08))*((t)-(&G000000001910288069E-08)))
(100.06517977)-(((t)*(4.9034306))*((t)-(10.1971193)))

Generation 1799, Farm 125, MDD 58, Fitness 13.278210306689259:
(&G000000019006581022E-08)-(((t)*(&G000000000499353961E-08))*((t)-(&G000000001910288069E-08)))
(100.06518022)-(((t)*(4.90343061))*((t)-(10.1971193)))

Generation 1814, Farm 125, MDD 73, Fitness 13.27821030668906:
(&G000000019006581024E-08)-(((t)*(&G000000000499353961E-08))*((t)-(&G000000001910288069E-08)))
(100.06518024)-(((t)*(4.90343061))*((t)-(10.1971193)))

Generation 1815, Farm 125, MDD 74, Fitness 13.278210306667496:
(&G000000019006581522E-08)-(((t)*(&G000000000499353961E-08))*((t)-(&G000000001910288069E-08)))
(100.06518577)-(((t)*(4.90343061))*((t)-(10.1971193)))

Generation 1824, Farm 125, MDD 83, Fitness 13.278210306667365:
(&G0000000019006581422E-08)-(((t)*(&G0000000000499353961E-08))*((t)-(&G0000000001910288069E-08)))
(100.06518422)-(((t)*(4.90343061))*((t)-(10.1971193)))

Generation 1832, Farm 125, MDD 91, Fitness 13.278210306666985:
(&G0000000019006581452E-08)-(((t)*(&G0000000000499353961E-08))*((t)-(&G0000000001910288069E-08)))
(100.06518457)-(((t)*(4.90343061))*((t)-(10.1971193)))

Generation 1836, Farm 125, MDD 95, Fitness 13.278210306666846:
(&G0000000019006581482E-08)-(((t)*(&G0000000000499353961E-08))*((t)-(&G0000000001910288069E-08)))
(100.06518482)-(((t)*(4.90343061))*((t)-(10.1971193)))

Generation 1840, Farm 125, MDD 99, Fitness 13.278210306666832:
(&G0000000019006581481E-08)-(((t)*(&G0000000000499353961E-08))*((t)-(&G0000000001910288069E-08)))
(100.06518481)-(((t)*(4.90343061))*((t)-(10.1971193)))

Generation 1853, Farm 125, MDD 88, Fitness 13.27821030665133:
(&G0000000019006581402E-08)-(((t)*(&G0000000000499353961E-08))*((t)-(&G0000000001910288079E-08)))
(100.06518402)-(((t)*(4.90343061))*((t)-(10.19711929)))

Generation 1854, Farm 125, MDD 87, Fitness 13.278210306645047:
(&G0000000019006580022E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288069E-08)))
(100.06519977)-(((t)*(4.90343062))*((t)-(10.1971193)))

Generation 1861, Farm 125, MDD 80, Fitness 13.278210306626486:
(&G0000000019006581402E-08)-(((t)*(&G0000000000499353961E-08))*((t)-(&G0000000001910288076E-08)))
(100.06518402)-(((t)*(4.90343061))*((t)-(10.19711926)))

Generation 1866, Farm 125, MDD 75, Fitness 13.278210306626322:
(&G0000000019006581402E-08)-(((t)*(&G0000000000499353961E-08))*((t)-(&G0000000001910288075E-08)))
(100.06518402)-(((t)*(4.90343061))*((t)-(10.19711925)))

Generation 1884, Farm 125, MDD 57, Fitness 13.278210306623471:
(&G0000000019006580222E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288069E-08)))
(100.06519777)-(((t)*(4.90343062))*((t)-(10.1971193)))

Generation 1888, Farm 125, MDD 53, Fitness 13.278210306608132:
(&G0000000019006581402E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288075E-08)))
(100.06518402)-(((t)*(4.90343062))*((t)-(10.19711925)))

Generation 1897, Farm 125, MDD 44, Fitness 13.278210306597284:
(&G0000000019006581462E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288075E-08)))
(100.06518462)-(((t)*(4.90343062))*((t)-(10.19711925)))

Generation 1900, Farm 125, MDD 41, Fitness 13.27821030659126:
(&G0000000019006581492E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288075E-08)))
(100.06518497)-(((t)*(4.90343062))*((t)-(10.19711925)))

Generation 1906, Farm 125, MDD 35, Fitness 13.278210306583379:
(&G0000000019006581602E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288076E-08)))
(100.06518602)-(((t)*(4.90343062))*((t)-(10.19711926)))

Generation 1909, Farm 125, MDD 32, Fitness 13.278210306532223:
(&G0000000019006581902E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288075E-08)))
(100.06518997)-(((t)*(4.90343062))*((t)-(10.19711925)))

Generation 1932, Farm 125, MDD 51, Fitness 13.27821030652767:
(&G0000000019006581902E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288074E-08)))
(100.06518997)-(((t)*(4.90343062))*((t)-(10.19711924)))

Generation 1936, Farm 125, MDD 55, Fitness 13.278210306527198:
(&G0000000019006581902E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288073E-08)))
(100.06518997)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 1957, Farm 125, MDD 76, Fitness 13.278210306527141:
(&G0000000019006581901E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288073E-08)))
(100.06518998)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 1959, Farm 125, MDD 78, Fitness 13.278210306527063:
(&G0000000019006581900E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288073E-08)))
(100.06518999)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 1963, Farm 125, MDD 82, Fitness 13.278210306522345:
(&G0000000019006580402E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288073E-08)))
(100.06519597)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 1970, Farm 125, MDD 89, Fitness 13.278210306517451:
(&G0000000019006580702E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288073E-08)))
(100.06519202)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 1987, Farm 125, MDD 94, Fitness 13.278210306517085:
(&G0000000019006580712E-08)-(((t)*(&G0000000000499353962E-08))*((t)-(&G0000000001910288073E-08)))
(100.06519217)-(((t)*(4.90343062))*((t)-(10.19711923)))

```
Generation 1989, Farm 125, MDD 92, Fitness 13.278210306516026:
(&G000000019006580502E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519402)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 1996, Farm 125, MDD 85, Fitness 13.278210306515948:
(&G000000019006580782E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519282)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 2001, Farm 125, MDD 80, Fitness 13.278210306515781:
(&G000000019006580792E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519297)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 2002, Farm 125, MDD 79, Fitness 13.278210306515641:
(&G000000019006580642E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519357)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 2034, Farm 125, MDD 47, Fitness 13.278210306515625:
(&G000000019006580652E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519342)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 2054, Farm 125, MDD 33, Fitness 13.278210306515621:
(&G000000019006580663E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519336)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 2059, Farm 125, MDD 38, Fitness 13.278210306515616:
(&G000000019006580653E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519343)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 2063, Farm 125, MDD 42, Fitness 13.278210306515611:
(&G000000019006580667E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519332)-(((t)*(4.90343062))*((t)-(10.19711923)))

Generation 2094, Farm 125, MDD 73, Fitness 13.278210306515605:
(&G000000019006580660E-08)-(((t)*(&G000000000499353962E-08))*((t)-(&G000000001910288073E-08)))
(100.06519339)-(((t)*(4.90343062))*((t)-(10.19711923)))
```

13.4.12 Round 4, Simulation 3

```
Generation 1, Farm 41, MDD 40, Fitness 0.7674717945896409: (ArcTan(67099.5))-((Tan(((-62))/(3493670.3)))+(Cos(phi)))
Generation 5, Farm 41, MDD 44, Fitness 0.7674717571012898: (ArcTan(67999.5))-((Tan(((-62))/(3493670.3)))+(Cos(phi)))
Generation 7, Farm 41, MDD 46, Fitness 0.7593875971312457: (ArcTan(67099.5))-((Tan(((-62))/(495.42385)))+(Cos(phi)))
Generation 8, Farm 27, MDD 47, Fitness 0.5109918101224717: ArcCos(Cos(phi))
Generation 10, Farm 0, MDD 49, Fitness 0.2572627081796269: Exp((-ArcSin(Cos((-((-0.179))-(phi))))))
Generation 12, Farm 0, MDD 51, Fitness 0.2525277717958834: Exp((-ArcSin(Cos((-((-0.175))-(phi))))))
Generation 16, Farm 4, MDD 55, Fitness 0.241877231386222: (3.581491)^(Cos((phi)+(3.1)))
Generation 17, Farm 11, MDD 56, Fitness 0.1127823419718788: Exp((-ArcSin(Cos(phi))))
Generation 20, Farm 21, MDD 59, Fitness 0.08801012885042317: Exp(Tan((-Cos((Tan(234246.59))+(Abs(phi))))))
Generation 26, Farm 21, MDD 55, Fitness 0.08746004603870419: Exp(Tan((-Cos((Tan(234246.56))+(Abs(phi))))))
Generation 33, Farm 37, MDD 48, Fitness 0.07893775049334747: (Sin(65.5937))^(Tan(Cos((-phi))))
Generation 42, Farm 37, MDD 39, Fitness 0.06893775049334749: (Sin(65.5937))^(Tan(Cos(phi)))
Generation 43, Farm 11, MDD 38, Fitness 0.06185974323672467: Exp((-Tan(Cos(phi))))
Generation 49, Farm 45, MDD 32, Fitness 0.04645117886586737: (1.21245)/((Cos(phi))+(1.230346))
Generation 53, Farm 45, MDD 28, Fitness 0.04643969448469899: (1.21245)/((Cos(phi))+(1.230356))
Generation 63, Farm 45, MDD 22, Fitness 0.04216352797255023: (1.21245)/((Cos(phi))+(1.235346))
Generation 70, Farm 45, MDD 29, Fitness 0.04061352495698945: (1.21245)/((Cos(phi))+(1.240026))
Generation 72, Farm 45, MDD 31, Fitness 0.04058840323480725: (1.21245)/((Cos(phi))+(1.240346))
Generation 73, Farm 45, MDD 32, Fitness 0.04058820731078036: (1.21247)/((Cos(phi))+(1.240346))
Generation 74, Farm 45, MDD 33, Fitness 0.04057375586805737: (1.21245)/((Cos(phi))+(1.241346))
Generation 76, Farm 45, MDD 35, Fitness 0.04048720651740863: (1.21815)/((Cos(phi))+(1.241346))
Generation 77, Farm 45, MDD 36, Fitness 0.04048698203368365: (1.21805)/((Cos(phi))+(1.241346))
Generation 80, Farm 45, MDD 39, Fitness 0.04041693239202693: (1.22245)/((Cos(phi))+(1.24225))
Generation 81, Farm 45, MDD 40, Fitness 0.04028418586591224: (1.230346)/((Cos(phi))+(1.246346))
Generation 83, Farm 45, MDD 42, Fitness 0.04019031584001107: (1.234346)/((Cos(phi))+(1.246347))
Generation 91, Farm 45, MDD 50, Fitness 0.04018976612094531: (1.246147)/((Cos(phi))+(1.250356))
Generation 92, Farm 45, MDD 51, Fitness 0.04014016411593398: (1.241146)/((Cos(phi))+(1.248026))
Generation 95, Farm 45, MDD 54, Fitness 0.04012814060321952: (1.25245)/((Cos(phi))+(1.250346))
Generation 99, Farm 45, MDD 58, Fitness 0.04012243131421026: (1.250358)/((Cos(phi))+(1.250356))
Generation 105, Farm 45, MDD 56, Fitness 0.04012162570030745: (1.250658)/((Cos(phi))+(1.250356))
Generation 106, Farm 45, MDD 55, Fitness 0.04012047696445718: (1.250358)/((Cos(phi))+(1.250156))
Generation 111, Farm 45, MDD 50, Fitness 0.04012047473292088: (1.250355)/((Cos(phi))+(1.250156))
Generation 112, Farm 45, MDD 49, Fitness 0.04012047255453641: (1.250352)/((Cos(phi))+(1.250156))
Generation 113, Farm 45, MDD 48, Fitness 0.0401204697327044: (1.250348)/((Cos(phi))+(1.250156))
Generation 114, Farm 45, MDD 47, Fitness 0.04012045158090155: (1.250318)/((Cos(phi))+(1.250156))
Generation 117, Farm 45, MDD 44, Fitness 0.04012042989685183: (1.250248)/((Cos(phi))+(1.250156))
Generation 119, Farm 45, MDD 42, Fitness 0.04012037156945399: (1.250158)/((Cos(phi))+(1.250056))
Generation 122, Farm 45, MDD 39, Fitness 0.04012036810808398: (1.250156)/((Cos(phi))+(1.250056))
Generation 124, Farm 45, MDD 37, Fitness 0.04012035115578733: (1.250146)/((Cos(phi))+(1.250056))
Generation 125, Farm 45, MDD 36, Fitness 0.04012030683306324: (1.250118)/((Cos(phi))+(1.250056))
Generation 128, Farm 45, MDD 33, Fitness 0.0401202997342901: (1.250158)/((Cos(phi))+(1.250086))
Generation 129, Farm 45, MDD 32, Fitness 0.04012022404572136: (1.250055)/((Cos(phi))+(1.250056))
Generation 132, Farm 45, MDD 29, Fitness 0.04012019574513775: (1.250028)/((Cos(phi))+(1.250056))
Generation 133, Farm 45, MDD 28, Fitness 0.04012018635664634: (1.250018)/((Cos(phi))+(1.250056))
```

```
Generation 144, Farm 45, MDD 23, Fitness 0.04012018276670802: (1.250014)/((Cos(phi))+(1.250056))
Generation 148, Farm 45, MDD 27, Fitness 0.04012018108129563: (1.250014)/((Cos(phi))+(1.250066))
Generation 150, Farm 45, MDD 29, Fitness 0.04012017755907722: (1.250008)/((Cos(phi))+(1.250056))
Generation 151, Farm 45, MDD 30, Fitness 0.04012017503503633: (1.250005)/((Cos(phi))+(1.250056))
Generation 155, Farm 45, MDD 34, Fitness 0.04012017437346008: (1.250005)/((Cos(phi))+(1.250058))
Generation 157, Farm 45, MDD 36, Fitness 0.04012017420550781: (1.250004)/((Cos(phi))+(1.250056))
Generation 163, Farm 45, MDD 42, Fitness 0.04012017256417844: (1.250002)/((Cos(phi))+(1.250056))
Generation 170, Farm 45, MDD 49, Fitness 0.04012017175237759: (1.250001)/((Cos(phi))+(1.250056))
Generation 172, Farm 45, MDD 51, Fitness 0.04012017126407373: (1.250001)/((Cos(phi))+(1.250058))
Generation 191, Farm 45, MDD 50, Fitness 0.04012017115386538: (1.250001)/((Cos(phi))+(1.250059))
```