PLOS ONE

RESEARCH ARTICLE

Inferring Mathematical Equations Using Crowdsourcing

Szymon Wasik[1,2¤]*, **Filip Fratczak**[1☯], **Jakub Krzyskow**[1☯], **Jaroslaw Wulnikowski**[1☯]

1 Institute of Computing Science, Poznan University of Technology, Poznan, Poland, **2** European Center for Bioinformatics and Genomics, Poznan University of Technology, Poznan, Poland

☯ These authors contributed equally to this work.
¤ Current Address: ul. Piotrowo 2, 60-965 Poznan, Poland
* szymon.wasik@cs.put.poznan.pl

Citation: Wasik S, Fratczak F, Krzyskow J, Wulnikowski J (2015) Inferring Mathematical Equations Using Crowdsourcing. PLoS ONE 10(12): e0145557. doi:10.1371/journal.pone.0145557

Editor: Chris T. Bauch, University of Waterloo, CANADA

Received: September 2, 2015

Accepted: December 4, 2015

Published: December 29, 2015

Data Availability Statement: All relevant data are within the paper and its Supporting Information files.

Funding: This study was partially supported by the Polish National Science Center under grant 2012/05/B/ST6/03026 (https://www.ncn.gov.pl/, SW) and the Polish National Center for Research and Development under grant LIDER/004/103/L-5/13/NCBR/2014 (http://www.ncbir.pl/, FF, JK, JW). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

Abstract

Crowdsourcing, understood as outsourcing work to a large network of people in the form of an open call, has been utilized successfully many times, including a very interesting concept involving the implementation of computer games with the objective of solving a scientific problem by employing users to play a game—so-called crowdsourced serious games. Our main objective was to verify whether such an approach could be successfully applied to the discovery of mathematical equations that explain experimental data gathered during the observation of a given dynamic system. Moreover, we wanted to compare it with an approach based on artificial intelligence that uses symbolic regression to find such formulae automatically. To achieve this, we designed and implemented an Internet game in which players attempt to design a spaceship representing an equation that models the observed system. The game was designed while considering that it should be easy to use for people without strong mathematical backgrounds. Moreover, we tried to make use of the collective intelligence observed in crowdsourced systems by enabling many players to collaborate on a single solution. The idea was tested on several hundred players playing almost 10,000 games and conducting a user opinion survey. The results prove that the proposed solution has very high potential. The function generated during weeklong tests was almost as precise as the analytical solution of the model of the system and, up to a certain complexity level of the formulae, it explained data better than the solution generated automatically by Eureqa, the leading software application for the implementation of symbolic regression. Moreover, we observed benefits of using crowdsourcing; the chain of consecutive solutions that led to the best solution was obtained by the continuous collaboration of several players.

Introduction

Among the many interesting trends associated with the development of technology, we can distinguish two that result directly from its popularization. First, the rapid increase in the number of computers, mobile devices, sensors, and other electronics has caused an

exponential increase in the amount of data collected, which, unfortunately, is not accompanied by an equally rapid development in techniques for data processing and knowledge discovery [1, 2]. Second, increasingly more people are wasting increasingly more time on the Internet performing activities that are not essential to life, coincidentally contributing to the data generation process. For example, according to 2012 research [3] approximately 32% of Facebook members use it to waste time, and 10% use it to access applications integrated within the portal. Considering that 89% of people with Internet access at work waste at least 30 min daily on tasks unrelated to their jobs, spending 23% of this time on Facebook [4], and including the fact that there are three billion Internet users, this adds up to millions of hours wasted on the Internet. If someone could utilize even a minute fraction of this human time to analyse data gathered on the Internet by exploiting the technique used by many crowdsourcing services, it could bring many benefits, especially considering that, as observed by Francis Galton in 1907, the collective opinion of a crowd of individuals can be much more precise than the opinion of a single expert [5].

The trends described above are probably the main reason for the rapid growth in the popularity of crowdsourcing. The term "crowdsourcing" was introduced in 2006 by Jeff Howe [6]. However, the concept of crowdsourcing, understood as outsourcing work to a large network of people in the form of an open call, is quite old. One of its first applications was when the British government offered prizes for the discovery of a method for measuring a ship's longitude in 1714 [7]. Since that time, the concept of crowdsourcing has been utilized many times, but the rapid progression of such techniques started with the development of the Internet in the 1990s. The best examples of its results are services such as Wikipedia or OpenStreetMap. An extended review of crowdsourcing systems on the WWW can be found in [8] and a discussion about the nature of crowdsourcing can be found in [9].

An interesting application of the crowdsourcing concept is in the implementation of computer games whose objective is to solve some scientific problem by playing the game. These games, which are called crowdsourced serious games [10], already have several uses in the solution of problems from various fields of science. One of the first successful attempts to implement a crowdsourced serious game was Foldit, which asks players to fold a protein to obtain its tertiary structure. Players are awarded points according to the number of biological and chemical constraints satisfied. The results obtained by the players were so good that they have even been collectively included as co-authors of the paper published in *Nature* as "Foldit Players" [11]. Another biological application, EteRNA, asks players to design RNA sequences that fold into a target shape. This game proves that both the community of players and the algorithm created by analysing their behaviour could outperform all other known methods [12]. Another useful example of a crowdsourcing game is Eyewire, whose objective is to map a network of connections between neurons based on microscopy data. The project has already achieved initial success by detecting areas that respond to localized motion [13]. Finally, there are games developed by Verigames.com that aim to use crowdsourcing to perform formal verification of the software. Among the games available from Verigames.com, the most interesting from the perspective of our research is Xylem, which attempts to use players with low mathematical skills to define loop invariants that are in fact mathematical formulae. Unlike most such games, Xylem is available not only for the PC but also as a mobile game for the iPad [14].

The great success of crowdsourcing in so many disciplines suggests that it could also be successfully applied to the automatic discovery of models to explain data. For many centuries, scientists and mathematicians have attempted to find natural laws that explain the world surrounding them and models of various dynamic systems. Given the development of artificial intelligence, especially with regard to evolutionary algorithms [15, 16], there is little

wonder that these algorithms also found application in an automatic search for mathematical equations that model various systems. In particular, some algorithms have recently appeared that successfully utilize symbolic regression analysis [17] to find such equations. One of the most successful is the Eureqa software application [18], which has been successfully used to solve many industrial and scientific problems. Other implementations include GPTIPS, a genetic programming and symbolic data mining platform for MATLAB [19], and RGP, a genetic programming framework for R that supports symbolic regression [20]. There have also been some successful attempts to model dynamic systems represented by differential equations [21–23]. However, all the above approaches are limited to rather simple systems because of the high computational complexity of the methods implemented [24]. Moreover, these approaches often tend to discover overfitted models because the fitness function usually prioritizes models that are more accurate rather than those with lower complexity [25].

The main objective of the present study was to verify whether crowdsourcing could be successfully applied to the discovery of mathematical equations that explain data gathered from dynamic systems. To achieve this goal, we designed and implemented a game in which players attempt to design a spaceship that represents an equation that models a given system. Additional objectives during the design of the game were to prepare a game targeted at people without advanced mathematical backgrounds and to make use of the collective intelligence observed in these crowdsourced systems. The game was tested by several hundred players in almost 10,000 games. Finally, we compared the results with the approach based on symbolic regression and analysed the users' opinions of the game.

Materials and Methods

Application design

The game was designed as a web application, "Throw the hamster" (available at http://hamster.ovh/; example gameplay is presented in the S1 Video), which allows players to take part in the modelling of a dynamic system based on data integrated inside the game. The modelling is conducted by playing a simple game in which the player must design a spaceship for the hamster. Users can build new solutions from scratch or by modifying other users' work. The flight trajectory of the spaceship depends on the design proposed by the user, and its objective is to fly as close as possible to stars placed in the sky. The spaceship flies from left to right, with its x-coordinate representing time and stars representing data points collected from some experiment. A model of the system is represented by the design of the spaceship in the form of a tree whose nodes consist of upgrades added to the spaceship. The tree structure is directly mapped into the expression tree [26] of the mathematical equation that models the system. Players can add, modify and remove the nodes. Each element affects the final trajectory. An example tree is presented in Fig 1. Players can test the designed structure at any time by shooting the hamster in the spaceship. After the hamster's flight ends, the player is awarded points based on the accuracy of the equation. It is also possible to share solutions on social networks, which potentially encourages new players to visit the website.

If the tested solution turns out to be one of the best, it appears in the ranking list. Ranked models are publicly available for everyone to see. Moreover, anyone can use them for further modifications and, thus, incrementally construct a new solution based on them. As a result, players do not need to build their own models from scratch; they can improve the most accurate solutions, thus enabling them to boost the score creatively but with minimal work and time invested.

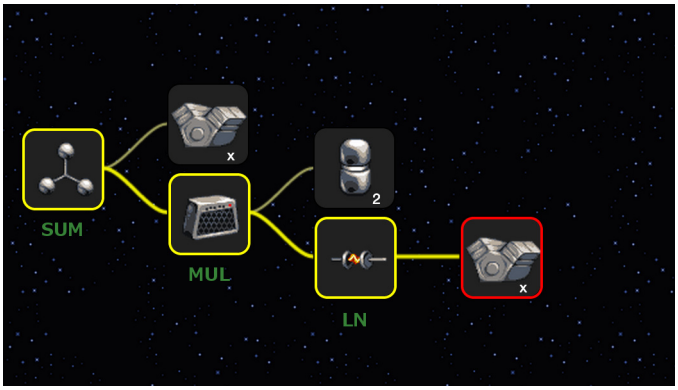

Fig 1. Example solution constructed by the player. The tree of upgrades to the spaceship represents the function $f(t) = t + 2\ln t$, which is the mathematical model of the system. In the figure x represents time (t in the function equation), and the value of the function models how the system behaves in time; however, the player does not have to know this. Green labels were added after the screenshot was taken to better explain how the solution was generated.

doi:10.1371/journal.pone.0145557.g001

Objective functions

Each solution prepared by players was evaluated via an objective function. For this purpose, we used a standard mean absolute error defined using the following formula:

$$score = \frac{1}{N}\sum_{i=1}^{N}|y_i - f(x_i)| \tag{1}$$

Here, N denotes the number of points (stars), y_i denotes the value of the data point recorded at time x_i, and $f(x_i)$ is the value predicted by the player's solution. As the solution becomes more accurate, the value of the objective function decreases, eventually reaching a very small value close to 0. The scoring procedure defined in Eq 1 is not very intuitive for users without mathematical backgrounds, so we proposed a transformation that guarantees three assumptions that are important from the perspective of usability. To make it easily understandable to players, the score should (1) increase as the solution improves, (2) not have negative values, and (3) be an integer. To guarantee these criteria, we defined a function that will always return a positive number less than or equal to 10,000:

$$result = \left\lfloor \frac{10000}{max(\sqrt{score}, 1)} \right\rfloor \tag{2}$$

Another function that characterizes the solution was used to measure its complexity and compare it with the results of the Eureqa software (described later). That is why we used the method of calculating the complexity of the mathematical formulae used in Eureqa. We assigned a weight to each of the operations that could appear in the formula and then summed the weights of all operations. The weights used by us are the same as the default weights used in the Eureqa software, and they are presented in Table 1. The value of the function that measured complexity was not presented to the players.

Table 1. Weights of particular operations used to measure the complexity of the solution.

Operation	Weight	Operation	Weight
addition	1	negation	1
subtraction	1	exponentiation	5
multiplication	1	logarithm	4
sine	3	natural logarithm	4
cosine	3	division	2
tangent	4	constant	1
cotangent	4	variable	1

doi:10.1371/journal.pone.0145557.t001

Testing

The testing procedure was conducted based on the data on Hepatitis C Virus (HCV) infections provided by Dahari et al. [27, 28]. These data describe how the viral RNA level decreases during therapy with pegylated interferon alpha and ribavirin. Because the analytical solution to the system of differential equations provided by Dahari is quite complex, we decided to attempt an automatic method to find the equation that explains the data.

Schedule and quantities. The tests were conducted using three iterations. After each iteration, the feedback from users was collected, the game design was analysed, bugs were fixed, and some new features were implemented. The statistics summarizing each iteration are presented in Table 2.

The preliminary iteration of tests (#0) consisted of internal tests on a small group of players. It helped to detect several bugs and a few problems related to the user experience. Statistics for this iteration are only estimations because bugs in the code and frequent database updates prevented their precise calculation. The first large-scale iteration (#1) was the first major test. The main objective of this iteration was to verify the proposed concept. The game was published on the Internet and presented to a wider group of people. We found several minor bugs, but, most importantly, we identified several misconceptions in the game's design. We corrected them and proceeded to the final iteration of tests (#2). These final results were also compared to the output of the Eureqa software. Details on the results are presented in the following sections.

The study involved Internet users who played the online game and were recruited by messages published on social networks. All statistics used during the research were collected anonymously, and all players were informed such that, according to Polish law, there was no need to collect consent from participants or obtain approval from the institutional review board. The authors did not have access to any potentially identifying information at any point of the study (including user IP addresses).

Reference results from Eureqa. We used version 0.99.9 of the Eureqa software, which was the most recent version when the experiment was conducted. The search was executed using

Table 2. Number of games, players and browser sessions in successive test iterations. Games played equals number of hamster launches. The time column presents the total time spent by all players on playing the game. The average duration of a single game is equal to 29.8 s.

Iteration	Players	Sessions	Games played	Games per user	Time
#0	∼20	∼40	∼400	∼20	∼3.3 *h*
#1	616	928	7,628	12.38	63.1 h
#2	90	212	1,525	16.94	12.6 h
Total	**726**	**1,180**	**9,553**	**13.16**	**79 h**

doi:10.1371/journal.pone.0145557.t002

the default values of parameters and the same mathematical operations, complexity weights and objective function as in the final iteration of the game (#2). The search was executed for 18 h using four cores of a 64-bit i7 CPU and, during that time, evaluated 1.5×10^{11} formulae.

Users opinion survey

During the two large test iterations, a survey was carried out to collect feedback on the game. The survey consisted of seven questions. We asked players how they liked the game and what they wanted to change about it. We also checked whether they were aware that this game was designed not only to be fun for the players but also to solve an important scientific problem. Finally, we asked about the players' attitudes toward mathematics to check whether there was a correlation between mathematical skills and the results in the game. Fifty-seven players out of 726 completed the survey. Detailed questions and all collected answers are presented in S1 Table, and results are discussed in the following sections.

Implementation

The main objective of the technical part of the project was to provide the application on the largest possible number of platforms. The game was implemented using the latest portable technologies, including HTML 5, CSS 3, Javascript, PHP and MySQL. Signing in to the game supports integration with Facebook and Google+ accounts and anonymous access. All screens are created according to a single-page-application pattern to provide a fluid user experience. The backend of the game stores all solutions in the database, which provides advanced analytical functions using the SQL language. Every solution has the structure of a tree, in which nodes are mathematical operations, variables or constant values. The tree is stored in JSON format, thus supporting interoperability.

Results

As presented in Table 2, during the first phase of tests, we attracted many more players. However, during the second phase, the players were much more dedicated to the game; on average, each of the players constructed 37% more solutions, so the results had higher quality instead of quantity. Some of the players in iteration #2 were repeats from the first phase; however, most of them were new players. Usually, players who constructed a small number of solutions did not find good formulae because the process of improving them is complex and requires time. The following sections contain a detailed analysis of each iteration of tests. Detailed results from all games that were played are presented in S2 Table.

Generated functions

Fig 2 presents the best functions from both the large-scale testing phases and the values of experimental data points. During the first test phase, players experimented significantly with trigonometric functions. Using trigonometric functions with a small period and large amplitude, it was possible to cover a wide range of points. The solution actually consisted of multiple overlapping sines and cosines. The objective function for such a solution has a high value, but a related modelling problem usually expects a simpler form of the equation. To solve this problem, we forbade the use of trigonometric functions during the second iteration of tests. After removing these functions, the best solution found by players was much simpler, and it does not overuse trigonometric functions.

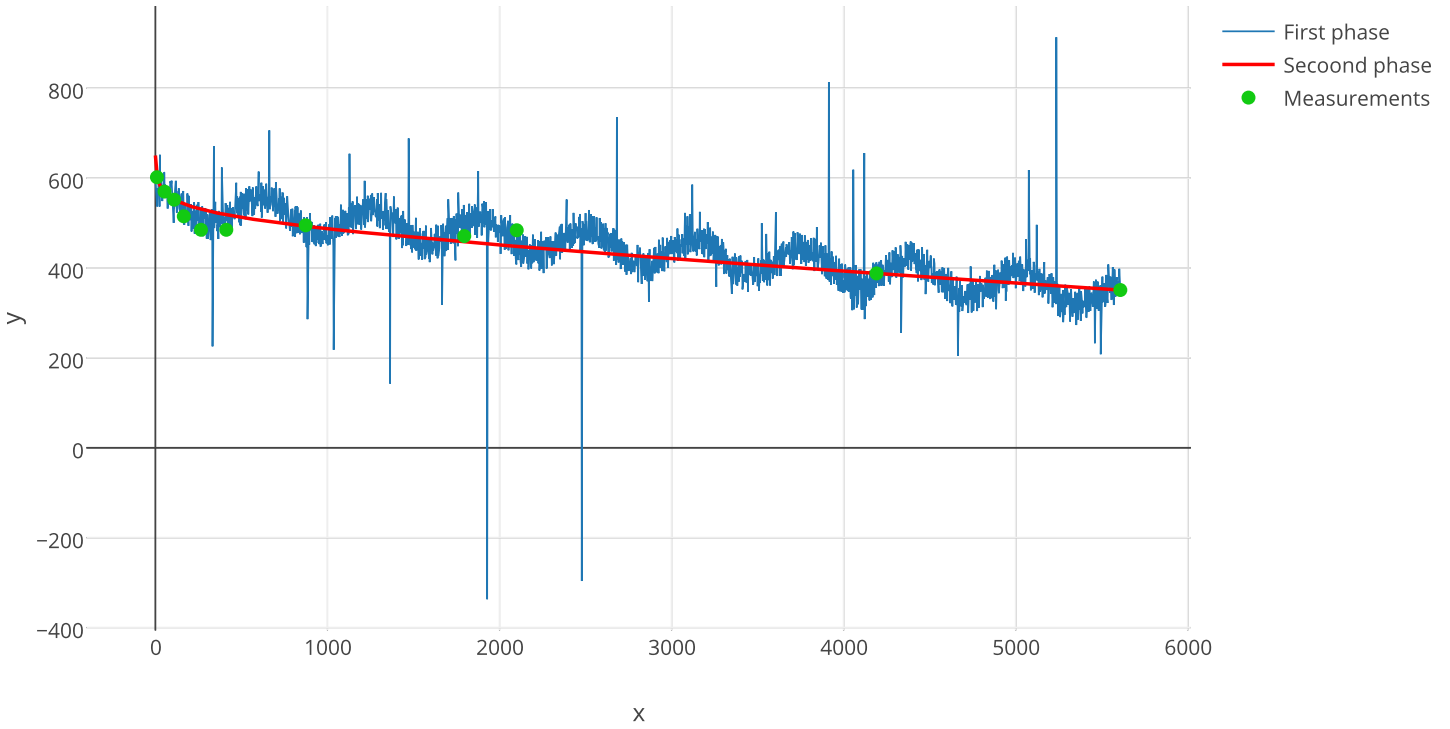

Fig 2. Comparison of the best functions obtained in each phase. In the first phase, the influence of trigonometric functions is easily visible. Moreover, when the denominator of the component containing the cosine function approaches 0, we can observe spikes in function values.

doi:10.1371/journal.pone.0145557.g002

Comparison with Eureqa

The following contains a comparison of the best functions constructed by players in each test iteration with results found by the Eureqa software. Eureqa does not find a single solution but a set of them; it stores the best solution for each level of formula complexity. From all of these solutions, we compared the best one with the best among all with a complexity less than or equal to the complexity of the solution constructed by players during the second phase of tests. For each function, we present the mean absolute error, the score presented to players in the game, and the complexity of the function. The functions can usually be simplified using basic arithmetic operations, but we avoided this to present the raw form of the formula generated by players or Eureqa.

Iteration #1: first testing phase. The best solution constructed by players. Mean absolute error: **4.93**. Game score: **4,510**. Weighted complexity: **65**.

$$\begin{aligned} f(x) = {} & 20 \cdot \sin\left(0.01 \cdot 20 \cdot \log\left(1 + x^{29.9969}\right) \cdot x \cdot (-0.1)\right) + (-0.036 \cdot x) + \\ & \frac{20 \cdot \cos\left(0.01 \cdot x\right)}{0.585} + 20 \cdot \sin\left(0.01 * 60.3\right) + 20 \cdot \cos\left(0.001 \cdot x^{2.98}\right) + \\ & 532.44 + \left(\frac{-24.8}{20 \cdot \cos\left(0.01 \cdot x^{1.555}\right)}\right) + 0.0051 \cdot \left(-1 \cdot 20 \frac{\log\left(1 + x\right)}{log(2)}\right) \end{aligned} \tag{3}$$

Iteration #2: second testing phase. The best solution constructed by players. Mean absolute error: **14.63**. Game score: **2,614**. Weighted complexity: **23**.

$$f(x) = 649.23 + \left(-1.019 \cdot 20 \frac{\log(1+x)}{\log(2.715)}\right) + (-0.022 \cdot x) + 1.13 + x \cdot x \cdot 0.0001 \cdot 0.0001 \cdot 0.2 \tag{4}$$

Eureqa #1. The best solution found by Eureqa with complexity less than or equal to the best solution from the second iteration of tests. Mean absolute error: **20.37**. Game score: **2,215**. Weighted complexity: **23**.

$$f(x) = 100\left(4.4 + \frac{3.5}{\frac{x}{100}+2} - 0.01\frac{x}{100} + 0.01 \cdot 4.4\right) \tag{5}$$

Eureqa #2. The best solution found by Eureqa. Mean absolute error: **5.93**. Game score: **4,104**. Weighted complexity: **72**.

$$f(x) = 100\left(2 + 4.4 - \sqrt{\sqrt{0.01 + \frac{x}{100}^{2} - \frac{x}{100} \cdot 2 \cdot 0.3^2} + \frac{\frac{x}{100}^{2} \cdot 0.3^2}{2^2}}\right) \tag{6}$$

The best result was obtained during the first phase of tests thanks to the use of trigonometric functions, which outperformed even the best solution generated by Eureqa. In a much more realistic case modelled during the second iteration of tests, the result is much worse; however, it is still slightly better than the result generated by Eureqa for the same complexity. Nevertheless, when not constrained by the complexity level, the result generated by Eureqa is better than the solution constructed by the players. The reason for this is probably the problem pertaining to the handling of complex solutions by players, which is later discussed in detail.

The time spent on calculating solutions in both approaches was comparable. Eureqa calculated the solution in approximately 18 h. In the second half of this time, improvements to the solution were very small, and at the end of this time, the solution stopped improving. Iteration #1, measured as the total time spent by all players on playing the game, was approximately three times longer and equal to approximately 63 h. However, in iteration #2, when more dedicated players were playing the game, the total time spent on solving the problem was a few hours shorter and equal to approximately 13 h. Details are presented in Table 2.

Gameplay analysis

Best solution formation. Figs 3 and 4 present how the value of the best solution changed over time as it was improved by players during the first and second phases of the tests. During each phase, there was a sharp increase in the value of the objective function over the first day of tests. During the first day of the first iteration, the solution was improved at four points, which are significantly connected with the addition of consecutive trigonometric functions. During the second phase of tests, it was more difficult to improve the solution without trigonometric functions; as a result, there are fewer improvements.

Highest increments. Most of the players constructed their solutions based on some other solution—their own solution or one of the best solutions constructed by other players. We

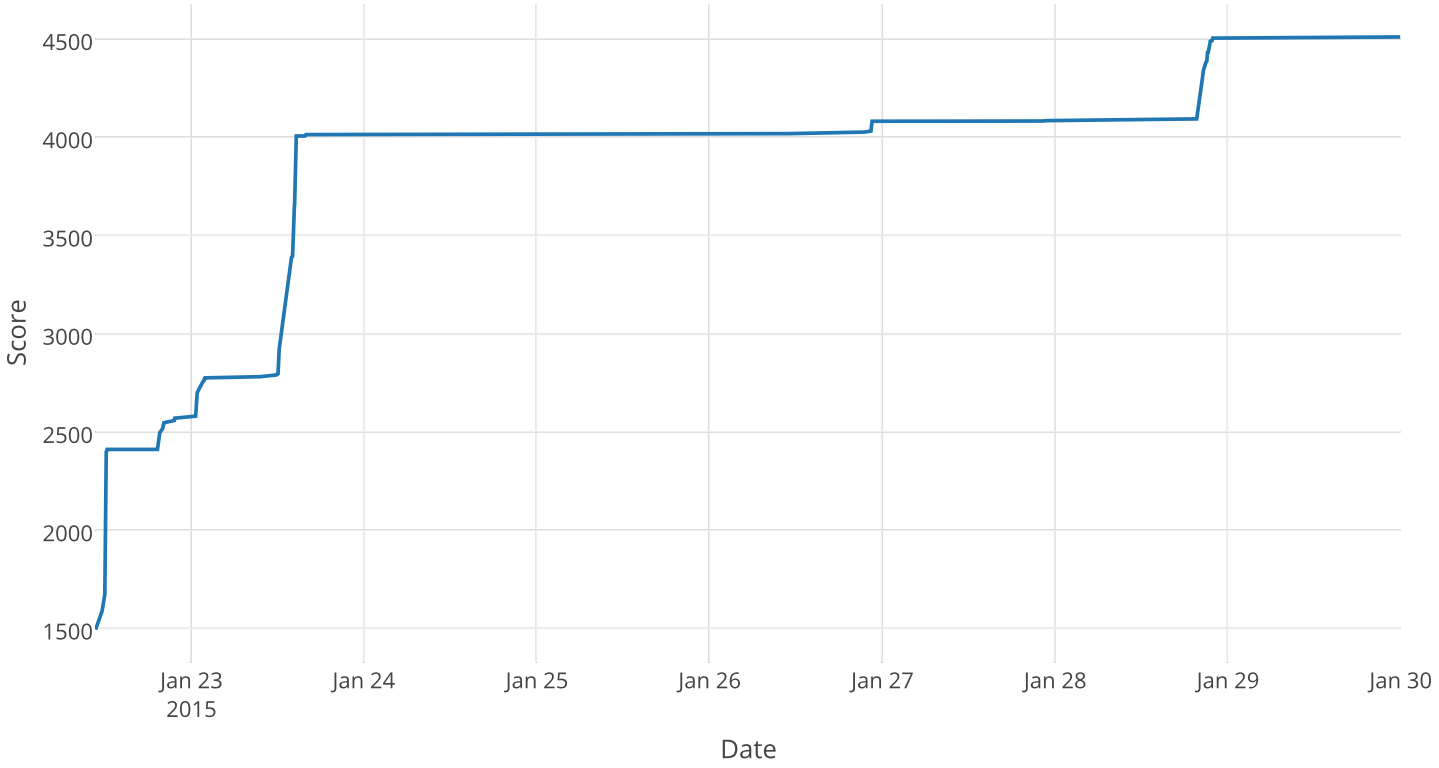

Fig 3. Formation of the best solution during the first testing phase.

doi:10.1371/journal.pone.0145557.g003

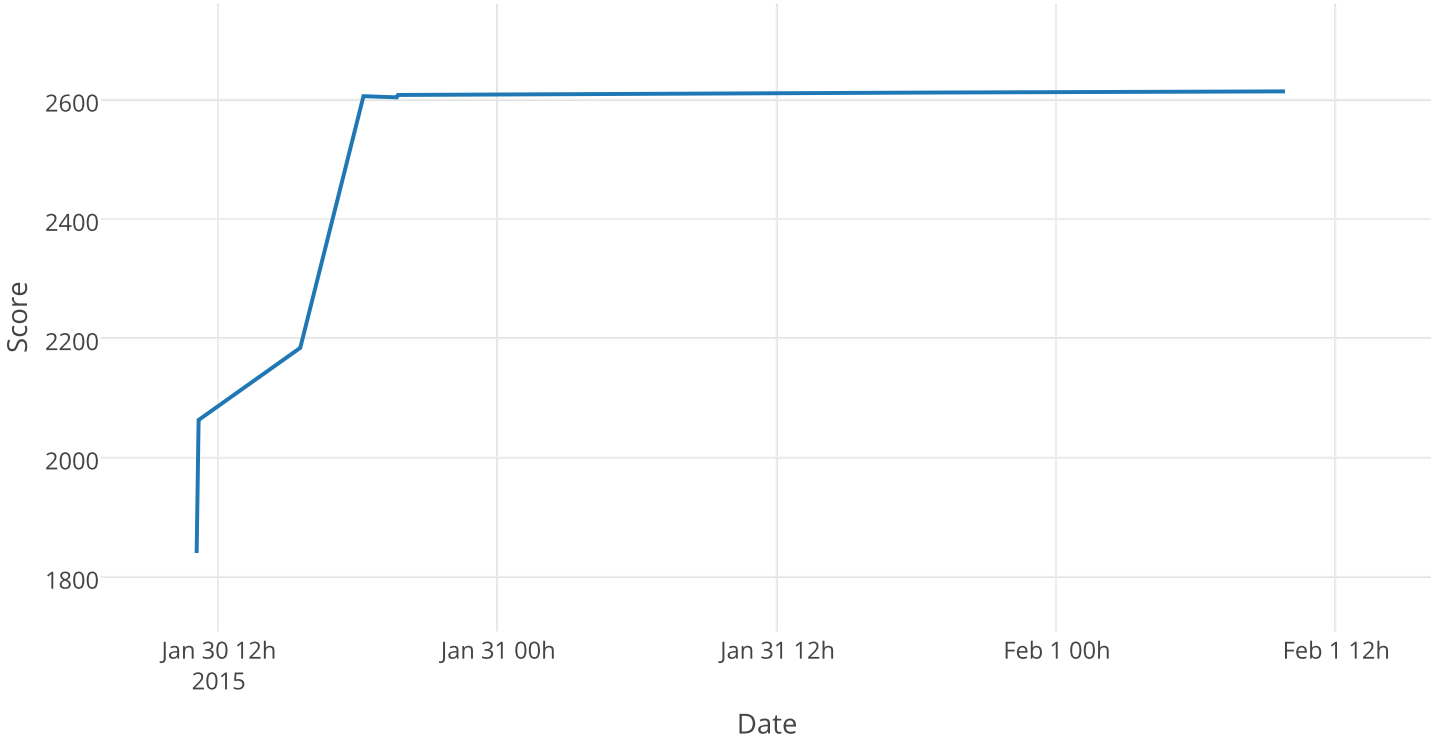

Fig 4. Formation of the best solution during the second testing phase.

doi:10.1371/journal.pone.0145557.g004

define an increment as the difference between the score of the new solution and the base score. Fig 5 presents the number of increments in each score range during the first phase of tests. The results for the second phase are similar. We present the results for the first phase because, during the second phase, fewer games were played.

Most increments were not high and were in the range of 0–100 points. Only a few players modified their functions in such a way as to achieve a result that was much higher than the base solution (with a score of at least 300 points higher). This is consistent with the analysis of the process on how the best solution was constructed (detailed data can be found in S3 Table). The best solution was created using 104 small improvements, usually by increments of fewer than 20 points, created by 17 different players. This is perfect proof of the collaborative nature of the process that led to the construction of the final solution.

Solution complexity. Fig 6 shows the best score for solutions constructed at each level of complexity during phase #1. The analysis of increment results for phase #2 was similar. For complexity in the range between 0 and 40, the value of the objective function is proportional to the complexity. This is quite obvious because increasing complexity allows the construction of more complex formulae that can be better fitted to data. For complexities between 40 and 70, the score does not change, and for complexities greater than 70, it is very difficult for users to design an effective solution. This is why, for larger complexities, Eureqa software outperforms the presented approach.

User opinion analysis

The survey confirmed that the appropriate game design and method for sharing information about the game ensured that people were aware of the scientific objective of the game. Eighty-

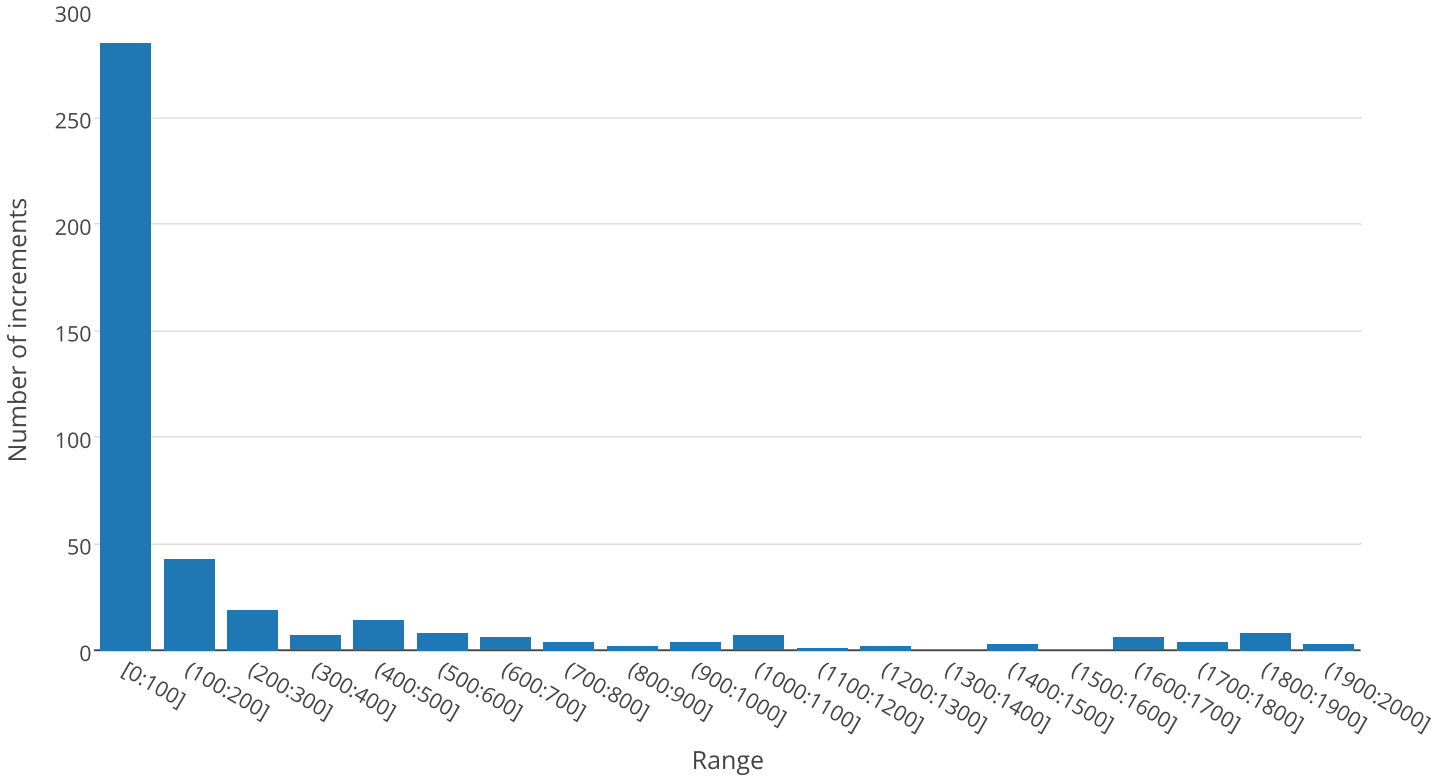

Fig 5. Number of score increments in each range.

doi:10.1371/journal.pone.0145557.g005

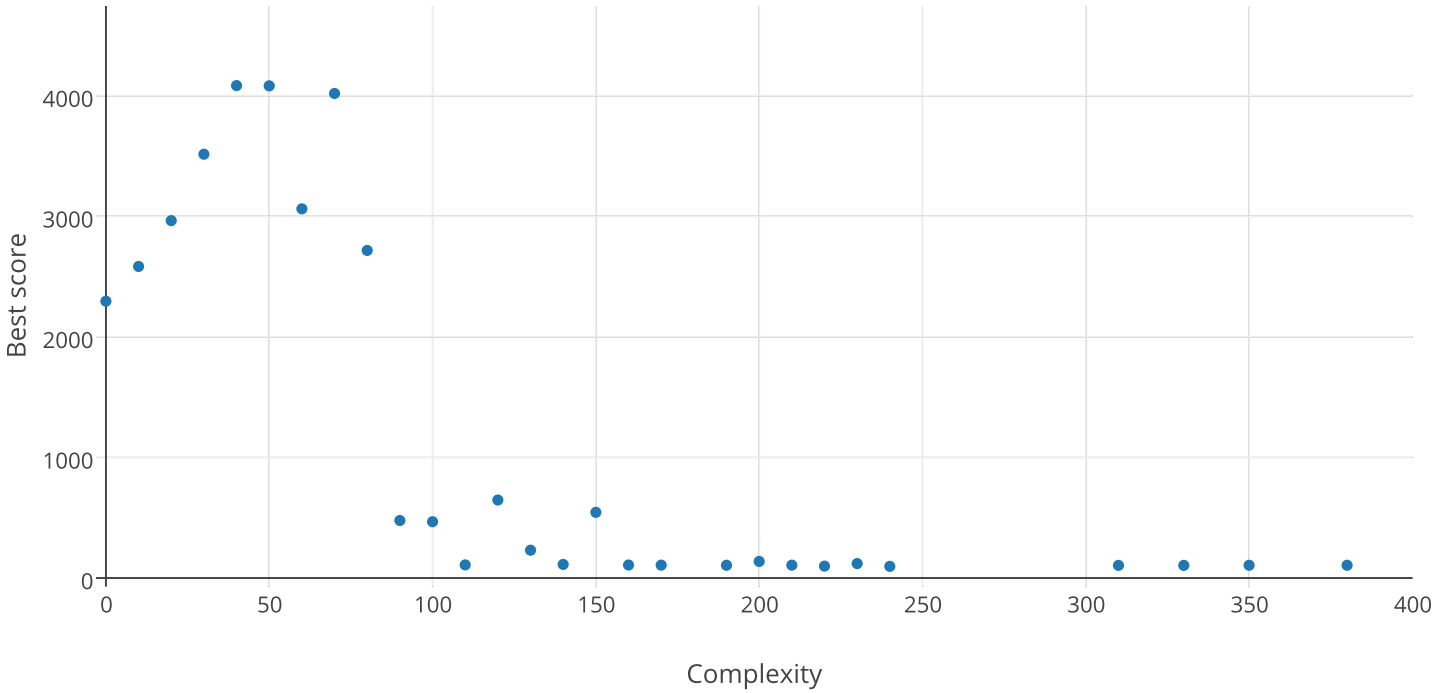

Fig 6. Best score for each solution complexity.

doi:10.1371/journal.pone.0145557.g006

two percent of them confirmed that they read the description explaining the scientific background of the game, and another 16% admitted that, although they had not read it, they were aware that there was some scientific aim. Many players declared that they played the game several times regardless of the fact that the game was not very interesting to them. This could suggest that they really understood the significance of the game and continued to play because of the scientific objective. Moreover, most players were more likely to recommend the game to others than not.

The survey also allowed the formation of some ideas about improvements that could be introduced to the game. Many people complained that they did not understand how adding upgrades to the spaceship influences its flight. This problem was partially solved during the second phase of tests by adding a description to each of the upgrades, thus explaining its influence and significantly improving the reception of the game during the second phase. In this iteration, for each upgrade, we provided the mathematical operation that it represents and an explanation for people with lower mathematical knowledge'for example, *"Sum—aggregates behaviour of connected elements"*, *"Subtraction—amplifies difference in behaviour of two elements"*. According to the survey, we also failed to attract users with lower mathematical skills; 95% of players declared an interest in mathematics.

All collected opinions are presented in S1 Table.

Discussion

The objective of the research was to verify whether it is possible to use a crowdsourced game to solve the problem of finding mathematical formulae to explain experimental data. To answer this question, we implemented a simple web game and integrated it with social networks. Based on the large number of games (almost 10,000), we can conclude that the verification was successful. The group of people could, in a relatively short time, construct a solution better

than that found by the leading software application that uses artificial intelligence algorithms based on symbolic regression. However, it should be noted that both solutions work only for the discovery of a formula of a single equation. Their application would be much more interesting if they could find formulae of a set of equations or, even better, a set of differential equations. Thus, it can be more easily used to explain how the system described by these equation works in reality. Nevertheless, these are much more complex problems that require separate consideration.

However, the artificial intelligence methods performed better for very complex formulae. The reason for this is probably the problem with controlling complex solutions manually by humans. This can be clearly seen in Fig 6. There is a complexity limit of 40, at which humans encounter difficulty in constructing good solutions, and another limit equal to 70, at which it is almost impossible to successfully process such a complex formula. Most of the users did not even try to construct such a complex solution, which is probably another reason why the value of the score for a complexity higher than 70 decreases dramatically. This difficulty can probably be solved by decomposition of the problem to smaller sub-problems—e.g., to design each component of the target formula separately. However, this creates new difficulties because such sub-problems are not independent, and the solution of one sub-problem influences how another sub-problem should be solved. These new difficulties in turn require a change in the values of data points based on the solutions of various sub-problems and, as a result, could require significant modification of the game's design assumptions.

The key observed advantage of crowdsourcing is the collaboration of many players. The creation of the best solution was possible thanks to the cooperation of 17 players who constructed a chain of more than 100 improvements (see S3 Table). Some of them introduced just one improvement, and some of them introduced several improvements in a row; however, the most interesting is the contribution of users 09, 11 and 14. They played the game several times, each of which improved the solution based on other players' contributions. The whole set of their actions presents behaviour similar to onsite collaboration in which several people are working together locally. However, thanks to online crowdsourcing, they did not have to be simultaneously collocated or even communicate with one another. The efficiency of the collaboration is also well presented in Fig 5, which proves that good solutions are usually constructed by a large number of small improvements created by different users.

Another interesting conclusion from the tests is the observation of how quickly players realized that using trigonometric functions could easily improve the solution's score. This was an obvious error in the game design that was successfully corrected before the second phase of tests. It is also worth noting that, for players, the biggest problem in the game was understanding how a change in the design of the spaceship influenced its flight trajectory. According to the user survey, it was a problem or a large problem for 40% of players. This was partially solved before the second phase of tests by adding a more detailed description of each upgrade; however, this solution can be improved further to obtain better results by including some type of tutorial.

One of the important conclusions from the tests and user opinion survey is that the game itself should be more interesting and engaging to attract players. The many volunteers that participated in the tests of the game have already proved that the implemented approach could be successful from a scientific point of view. However, they emphasized that the game should be more attractive to players to stimulate them to play the game for a longer time. This is why we have currently suspended the search for new players and are preparing a new version of the game that will be more interesting for users to play while addressing the most serious concern observed during the tests—difficulties in understanding how the design of the spaceship influences its flight.

To summarize the article, we presented a novel approach for finding mathematical formulae that explain experimental data gathered from the analysis of dynamic systems. The solution is a crowdsourced serious game, which proved to be very successful in solving this problem. There are still some drawbacks that must be solved before widespread implementation of this method, but they were identified during the research and well defined, and we have some ideas on how to solve them. Currently, the game can be classified as a very difficult puzzle game, but adding some action elements to the simulation of the hamster's flight could make it much more entertaining. The best proof of how a minimal, score-based action game can engage millions of players can be provided, for example, by the success of the simple Flappy Bird game [29]. After solving the identified drawbacks, it would also be interesting to test the game based on more datasets from various areas of science.

Supporting Information

S1 Video. Example gameplay. Video that demonstrates an example interaction with the game: construction of the spaceship, its flight and selecting another user's solution as a starting solution to improve the spaceship.
(MP4)

S1 Table. Survey results. The table contains all answers to the closed-ended questions inside the survey. It also contains the description of each question and possible answers. The information about the game was spread among our friends; that is why most of the open-ended questions were answered in Polish, so we do not include them. The analysis of answers to open-ended questions is included in the article.
(XLSX)

S2 Table. Game results. Game results generated during the first and the second iterations of tests. Each row presents the solution designed by the player, its value, the player's id and the base solution that was used.
(XLSX)

S3 Table. Chain of improvements. Chain of improvements that lead to the best solution. It presents the sequence of solutions, each of which is based on the previous one with the value of the increment and id of the player.
(XLSX)

Author Contributions

Conceived and designed the experiments: SW FF JK JW. Performed the experiments: FF JK JW. Analyzed the data: SW FF JK JW. Contributed reagents/materials/analysis tools: SW FF JK JW. Wrote the paper: SW FF JK JW.

References

1. Szalay A, Gray J. 2020 Computing: Science in an exponential world. Nature. 2006 mar; 440 (7083):413–414. Available from: http://dx.doi.org/10.1038/440413a. doi: 10.1038/440413a PMID: 16554783
2. Villars RL, Olofson CW, Eastwood M. Big data: What it is and why you should care. White Paper, IDC. 2011; Available from: http://www.emitac-ees.ae/wp-content/uploads/2014/04/IDC_AMD_Big_Data_Whitepaper.pdf.
3. Giannakos MN, Chorianopoulos K, Giotopoulos K, Vlamos P. Using Facebook out of habit. Behaviour & Information Technology. 2013 jun; 32(6):594–602. Available from: http://dx.doi.org/10.1080/0144929x.2012.659218. doi: 10.1080/0144929X.2012.659218

4. Gouveia A. 2014 Wasting Time at Work Survey [Blog]; 2014 [cited 2015.02.17]. Available from: http://www.salary.com/2014-wasting-time-at-work/slide/12/.
5. Galton F. Vox Populi. Nature. 1907 mar; 75(1949):450–451. Available from: http://dx.doi.org/10.1038/075450a0. doi: 10.1038/075450a0
6. Howe J. The Rise of Crowdsourcing. Wired. 2006 June; 14.06.
7. Dawson R, Bynghall S. The rise of crowdsourcing. In: Getting Results From Crowds: The definitive guide to using crowdsourcing to grow your business. Advanced Human Technologies Inc; 2011. p. 9–12.
8. Doan A, Ramakrishnan R, Halevy AY. Crowdsourcing systems on the World-Wide Web. Commun ACM. 2011 apr; 54(4):86. Available from: http://dx.doi.org/10.1145/1924421.1924442. doi: 10.1145/1924421.1924442
9. Estelles-Arolas E, de Guevara FGL. Towards an integrated crowdsourcing definition. Journal of Information Science. 2012 mar; 38(2):189–200. Available from: http://dx.doi.org/10.1177/0165551512437638. doi: 10.1177/0165551512437638
10. Tellioglu U, Xie GG, Rohrer JP, Prince C. Whale of a crowd: Quantifying the effectiveness of crowdsourced serious games. In: 2014 Computer Games: AI Animation Mobile, Multimedia, Educational and Serious Games (CGAMES). Institute of Electrical & Electronics Engineers (IEEE); 2014. p. 1–7. Available from: http://dx.doi.org/10.1109/cgames.2014.6934151.
11. Cooper S, Khatib F, Treuille A, Barbero J, Lee J, Beenen M, et al. Predicting protein structures with a multiplayer online game. Nature. 2010 aug; 466(7307):756–760. Available from: http://dx.doi.org/10.1038/nature09304. doi: 10.1038/nature09304 PMID: 20686574
12. Lee J, Kladwang W, Lee M, Cantu D, Azizyan M, Kim H, et al. RNA design rules from a massive open laboratory. Proceedings of the National Academy of Sciences. 2014 jan; 111(6):2122–2127. Available from: http://dx.doi.org/10.1073/pnas.1313039111. doi: 10.1073/pnas.1313039111
13. Helmstaedter M, Briggman KL, Turaga SC, Jain V, Seung HS, Denk W. Corrigendum: Connectomic reconstruction of the inner plexiform layer in the mouse retina. Nature. 2014 oct; 514(7522):394–394. Available from: http://dx.doi.org/10.1038/nature13877. doi: 10.1038/nature13877
14. Logas H, Whitehead J, Mateas M, Vallejos R, Scott L, Shapiro D, et al. Software Verification Games: Designing Xylem, The Code of Plants. In: Foundations of Digital Games 2014 (FDG2014). Center for Games and Playable Media; 2014. p. 1–8. Available from: http://www.fdg2014.org/papers/fdg2014_paper_17.pdf.
15. Krawiec K, Wieloch B. Analysis of Semantic Modularity for Genetic Programming. Foundations of Computing and Decision Sciences. 2009; 34 ProjectELP(4):265–285.
16. Jaśkowski W, Krawiec K. Formal analysis, hardness, and algorithms for extracting internal structure of test-based problems. Evolutionary computation. 2011; 19(4):639–671. doi: 10.1162/EVCO_a_00046 PMID: 21815770
17. Willis MJ. Genetic programming: an introduction and survey of applications. In: Second International Conference on Genetic Algorithms in Engineering Systems. Institution of Engineering and Technology (IET); 1997. p. 314–319. Available from: http://dx.doi.org/10.1049/cp:19971199.
18. Schmidt M, Lipson H. Distilling Free-Form Natural Laws from Experimental Data. Science. 2009 apr; 324(5923):81–85. Available from: http://dx.doi.org/10.1126/science.1165893. doi: 10.1126/science.1165893 PMID: 19342586
19. Searson DP, Leahy DE, Willis MJ. GPTIPS: an open source genetic programming toolbox for multigene symbolic regression. In: Proceedings of the International multiconference of engineers and computer scientists. vol. 1. Citeseer; 2010. p. 77–80.
20. Cortez P. Population Based Search. In: Modern Optimization with R. Use R!. Cham: Springer International Publishing; 2014. p. 63–98. Available from: http://link.springer.com/10.1007/978-3-319-08263-9.
21. Gaucel S, Keijzer M, Lutton E, Tonda A. Learning Dynamical Systems Using Standard Symbolic Regression. In: Genetic Programming. Springer Berlin Heidelberg; 2014. p. 25–36. Available from: http://dx.doi.org/10.1007/978-3-662-44303-3_3.
22. Cornforth T, Lipson H. Symbolic regression of multiple-time-scale dynamical systems. In: Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference—GECCO '12. Association for Computing Machinery (ACM); 2012. p. 735–742. Available from: http://dx.doi.org/10.1145/2330163.2330266.
23. Schmidt M, Vallabhajosyula R, Jenkins J, Hood J, Soni A, Wikswo J, et al. Automated refinement and inference of analytical models for metabolic networks. Phys Biol. 2011; 8:055011. PMID: 21832805
24. Daniels BC, Nemenman I. Automated adaptive inference of coarse-grained dynamical models in systems biology. arXiv preprint arXiv:14046283. 2014;.

25. Affenzeller M, Winkler SM, Kronberger G, Kommenda M, Burlacu B, Wagner S. Gaining deeper insights in symbolic regression. In: Genetic Programming Theory and Practice XI. Springer New York; 2014. p. 175–190.

26. Aho AV, Johnson SC. Optimal Code Generation for Expression Trees. JACM. 1976 jul; 23(3):488–501. Available from: http://dx.doi.org/10.1145/321958.321970. doi: 10.1145/321958.321970

27. Dahari H, Ribeiro RM, Perelson AS. Triphasic decline of hepatitis C virus RNA during antiviral therapy. Hepatology. 2007; 46(1):16–21. Available from: http://dx.doi.org/10.1002/hep.21657. doi: 10.1002/hep.21657 PMID: 17596864

28. Wasik S, Jackowiak P, Figlerowicz M, Blazewicz J. Multi-agent model of hepatitis C virus infection. Artificial Intelligence in Medicine. 2014 feb; 60(2):123–131. Available from: http://dx.doi.org/10.1016/j.artmed.2013.11.001. doi: 10.1016/j.artmed.2013.11.001 PMID: 24309221

29. Isaksen A, Gopstein D, Nealen A. Exploring game space using survival analysis. Foundations of Digital Games. 2015;.