Data-driven partial differential equations discovery approach for the noised multi-dimensional data

Mikhail Maslyaev, Alexander Hvatov, and Anna Kalyuzhnaya

ITMO University, Kronsersky pr. 49, 197101, St. Petersburg, Russia
alex_hvatov@corp.ifmo.ru

Abstract. Data-driven methods provide model creation tools for systems, where the application of conventional analytical methods is restrained. The proposed method involves the data-driven derivation of a partial differential equation (PDE) for process dynamics, which can be helpful both for process simulation and studying. The paper describes the progress made within the PDE discovery framework. The framework involves a combination of evolutionary algorithms and sparse regression. Such an approach gives more versatility in comparison with other commonly used methods of data-driven partial differential derivation by making fewer restrictions on the resulting equation. This paper highlights the algorithm features which allow the processing of data with noise, which is more similar to the real-world applications of the algorithm.

Keywords: data-driven modelling · PDE discovery · evolutionary algorithms · sparse regression · spatial fields · physical measurement data

1 Introduction

The increasing quality and quantity of measurement techniques and the emerging reliable sets of high-resolution data of physical processes in the current years give a hand to the advances of data-driven modelling (DDM). The ability to simulate complex processes, neglecting a lack of knowledge about the underlying structure of the system, can be vital for the development of models in such spheres of science as biology, medicine, materials technology, and metocean studies. In general, data-driven modelling involves the development of complete models from various fields of measurements, using means of statistics and machine learning algorithms. However, on some occasions, DDM can enhance the existing models with the addition of supplementary expressions or by a refinement of weight values [1]. In the fluid dynamics science and hydrometeorology, development of surrogate models is the most common application of data-driven algorithms.

The majority of modern data-driven algorithms of equation derivation are based on the artificial neural networks [4, 8, 9]. However, this approach in the majority of cases leads to the non-interpretable models. It means that the model that is represented by the neural network acts as the black box, while the results of the prediction or the modeling overall may be used to solve the particular problem.

On the contrary, the regression models may be interpretable, while the problem is solved with moderate quality. In this paper, the approach that involves the derivation of governing differential equations for the dynamic system is described. This approach may be considered as the compromise between the quality of the neural networks and the simplicity of the pure regression models. The predictions for the future state of the system can be acquired by solving the resulting equation. This statement of the problem corresponds with the specifics of the metocean study, in which the systems can usually be defined by single or series of partial differential equations.

In contrast to the other similar algorithms, which also use sparse regression [2, 10–12], the proposed one does not require the construction of token libraries, from which the equation terms are selected. The developed method is focused on the construction of terms from simple tokens that contain the original function and its derivatives of orders selected in a specified range. Additionally, unlike other groups of methods, primarily presented by artificial neural networks, the proposed method does not assume the presence of the first-order time derivative, giving it both versatility in selecting the type of time dynamics and the opportunity to select the steady-state of a system, if it fits data the best. In the current work, the algorithm for a single equation case is developed.

The previous works [5] have shown the ability of the described method to derive the correct structure of the equation for the case of the evolution equations with one spatial coordinate, even when the noise is added, and for the synthetic noiseless case of the two-dimensional wave equation. In this paper, the algorithm is extended to the cases with higher dimensionality and for the noised input fields. This problem statement mimics one of the possible application of the framework, that is connected with the derivation of governing equations for data-driven modelling of the fluids dynamics, that are connected with hydrometeorological studies.

In the paper, the particular problem of the noise in the higher-dimensional data is considered. Usually, the fact that the differentiation error increases exponentially when the dimensionality of the problem grows is ignored in the references, and only the one-dimensional cases are considered. Paper is organised as follows: in Sec. 2 the problem of the data-driven PDE discovery is briefly described. Sec. 3 presents the additions to the method described in the previous article [5] that allow dealing with the higher-dimension data-driven PDE discovery. In Sec. 4, numerical examples on the synthetic data, as well as on the real data, are shown. Sec. 5 concludes the paper.

2 Problem statement

The class of problems, which can be solved by the described method, can be summarized as follows: the process, which involves scalar field u, occurring in the area Ω, is governed by the partial differential Eq. 1. However, there is no a priori information about the dynamics of the process except the fact, that it can be described by some form of PDE (for simplicity, we consider temporally varying 2D field case, however, the problem could be formulated for an arbitrary field).

$$\begin{cases} F(u, \frac{\partial u}{\partial x_1}, \frac{\partial u}{\partial x_2}, ..., \frac{\partial u}{\partial t}, \frac{\partial^2 u}{\partial x_1^2}, \frac{\partial^2 u}{\partial x_2^2}, ..., \frac{\partial^2 u}{\partial t^2}, ...) = 0; \\ G(u) = 0, \ u \in \Gamma(\Omega) \times [0, T]; \end{cases} \tag{1}$$

From the area $\Omega \times [0, T]$ a set of samples $\mathrm{U} = \{u_1, u_2, ..., u_n\}$, where $u_i = u(x_1^{(i)}, x_2^{(i)}, t_i)$ is the function value at the arbitrary point $(x_1^{(i)}, x_2^{(i)}, t_i) \in \Omega \times [0, T]$, is collected. There are no strict limitations for the distribution of the sample collection points in the area, but the further requirement of the derivative calculation makes the case of stationary points, located on the grid, more preferable than any other situation. The main task of the algorithm is the derivation of the Eq. 1, using measurements from the set of discrete measurements U with some externally defined limitations, including a range of the derivative orders, a number of terms in the equations and number of factors in the term.

The noise in this paper is assumed to be directional, where the noise in the direction x_j can be described as

$$E_{x_j}(u(x_1, ..., x_n; t)) = u(\bar{x_1}, ..., x_j, ...\bar{x_n}; t) + \epsilon(x_j) \tag{2}$$

With $\bar{x_i}$ "fixed" variables are denoted and $\epsilon(x_j)$ is the noise, which in the paper assumed to be distributed normally $N(0; \sigma)$ with the expected value of 0 and the variance of σ. It should be emphasized that poly-directional noise forms as the superposition of the unidirectional noise operators, i.e. $E_{x_j, x_k} = E_{x_j} \circ E_{x_k}$. In what follows $\bar{\sigma} = \sigma$ max(u) is chosen as the multiplier of maximal magnitude of the measured value in this direction. The noise level is defined in the same way. In the text below bar over the variance, σ is omitted.

3 Method description

In this section, the details of the evolutionary method of partial differential equation derivation are described. The proposed method involves a combination of evolutionary algorithm and sparse regression for the detection of the equation structure. The former is aimed at the construction of equation terms set, while the latter is focused on the selection of significant terms from the created set and calculation of weights that will be present in the resulting equation.

3.1 Data preprocessing

To initialize algorithm, time, and spatial derivatives, which will later form the desired equation, must be calculated. In specific situations, the derivatives by themselves can be measured and, therefore, this step can be skipped, but often, only the raw value of the studied function is available for the research. For the more straightforward further computations, it can be assumed without loss of the generality, that the measurements are held on the rectangular (but not necessarily uniform) grid. Unlike the instances of single space dimension, in some experiments, even on data with moderate noise levels, the finite-difference method of derivative calculation can lead to satisfactory results, the multi-dimensional case requires advanced methods of obtaining derivatives. It is important to note that the quality of the taken derivatives is crucial for acquiring the correct structure of the equation. In some cases, when the data has high noise values, the error of the equation $||F||$ can be lower with an utterly incorrect set of terms.

In general, the calculation of derivatives is the operation that is vulnerable to the noise in the data. On the other hand, the result of the algorithm, in general, depends on the quality of the input derivatives: if they are computed with high errors, the alterations of the resulting structure of the equation can vary from incorrect coefficients to the entirely wrong structure. For these reasons, several noise-resistant methods of partial derivative calculations have been introduced. Notably, they include such commonly used methods as kernel smoothing [3], derivation of polynomials, fitted over sets of points, and more uncommon ones like Kalman filtering [7].

Therefore, in the framework, data clearance and noise-resistant derivative calculations have been combined to achieve decent smoothness. First of all, Gaussian smoothing kernels are applied for the data field in each time frame. This approach can reduce the significant outliers in the data and corresponds to the nature of the studied metocean processes, where the fields tend to be smooth. In the case of the time-dependent two-dimensional field, the smoothing is applied for each of the time frames. Two-dimensional Gaussian smoothing with selected bandwidth σ has the structure Eq. 3 and kernel Eq. 4, where s is the point, for which the smoothing is done, and s' - point, that value is utilized in smoothing.

$$\tilde{u}(s,t) = \int K_\sigma(s-s')u(s')ds'; \tag{3}$$

$$K_\sigma(s-s') = \frac{1}{2\pi\sigma^2}\exp\left(\frac{1}{2\sigma^2}\sum_{i=1}^{2}(s-s')_i\right); \tag{4}$$

In addition to smoothing, a noise-stable numerical differentiation scheme is applied. The derivative is taken by differentiation of polynomials, constructed over the set of points in some window. The coefficients of the polynomials, utilized in this step, are defined by linear regression. Despite all these measures, as it is presented on Tab. 1, derivatives of higher orders tend to have significant errors even after smoothing and polynomial derivation.

A particular example of the noised measure fields is shown in Fig. 1. For the clarity matters, only the spatial field center slice is provided. However, it should be emphasized that the entire spatial field is smoothed out to obtain the spatial derivative field.

Table 1. Noise levels (%) for the raw noised data and for the smoothed data

	u	$\frac{\partial u}{\partial t}$	$\frac{\partial^2 u}{\partial t^2}$
Noised function	15.5	260.1	12973.8
Smoothed function	12.3	10.78	458.2

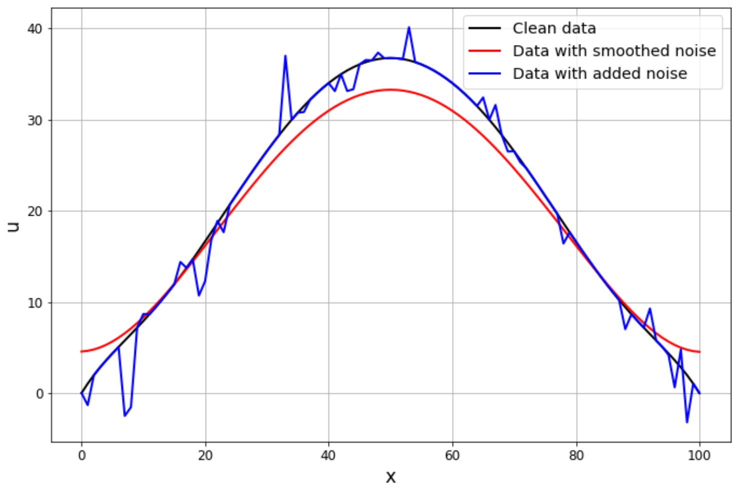

Fig. 1. Graph of a section over one spatial dimension for synthetic input function (solution of wave equation with 2 spatial dimensions) in original state, with added Gaussian noise, and after the noise was smoothed by Gaussian kernel

Differentiation of the three different fields shown in Fig. 1 gives the derivative fields that have values of the different orders. Thus, they are shown in the different graphs in Fig. 2.

In the majority of the real-world processes, derivative orders are limited to the first or second order. Derivative field of the slices shown in Fig. 1 represented in Fig. 2 show that the proposed algorithm of noise reduction in derivatives not only achieves values, close to the values of the derivatives on clear data, but also the structure of the fields are similar, which is vital for the evolutionary algorithm.

3.2 Evolutionary algorithm

After the differentiation process commits, the evolutionary algorithm is initiated. Individual derivatives are taken as tokens (Eq. 5), combinations of which form the terms of the searched equation. An example of such a combination is presented in

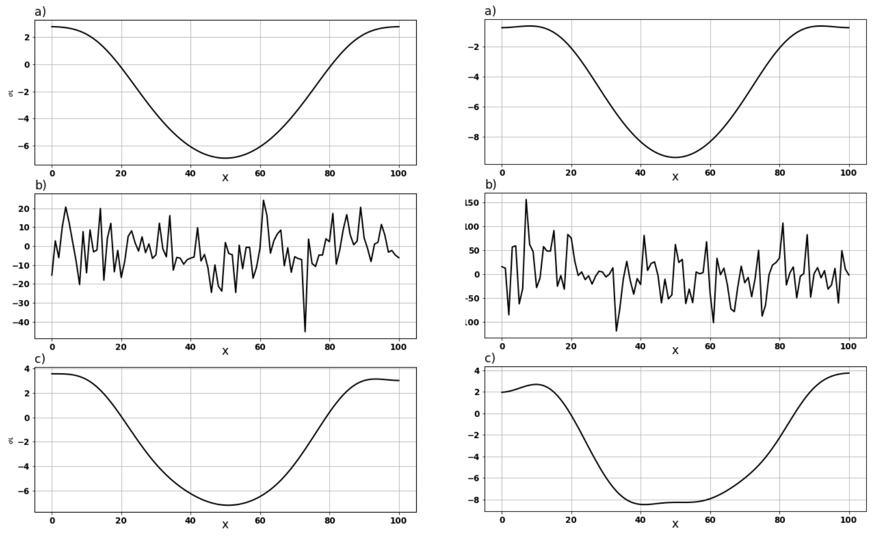

Fig. 2. Graph of first (left column) and second (right column) order time derivative, calculated on input function (a)), noisy function (b)), and function with noise, smoothed by Gaussian kernel (c)

the Eq. 6. The vector, that will after further modifications are used as the feature for the regression, is composed as the elementwise product (that is denoted with $\odot$ symbol) of vectors containing original function and its derivative along the x-axis.

$$\mathbf{f}_1 = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{bmatrix}; \mathbf{f}_2 = \begin{bmatrix} u\left(t_0,\ x_0\right) \\ \vdots \\ u\left(t_i,\ x_j\right) \\ \vdots \\ u\left(t_m,\ x_n\right) \end{bmatrix}; \mathbf{f}_3 = \begin{bmatrix} u_x\left(t_0,\ x_0\right) \\ \vdots \\ u_x\left(t_i,\ x_j\right) \\ \vdots \\ u_x\left(t_m,\ x_n\right) \end{bmatrix}; \ldots \tag{5}$$

$$\mathbf{F}'_k = \begin{bmatrix} u(t_0, x_0) * u_x(t_0, x_0) \\ \vdots \\ u(t_i, x_j) * u_x(t_i, x_j) \\ \vdots \\ u(t_m, x_n) * u_x(t_m, x_n) \end{bmatrix} = \mathbf{f_2} \odot \mathbf{f_3}; \tag{6}$$

The normalization of terms values is held for each of the time frames, passed into the algorithm, for the correct operation of regularized regression during the further weight calculation phases. In this step, various norms can be used, but most commonly, L_2 norm or L_∞ norms are applied, as it is represented in the Eq. 7.

$$\mathbf{F}_k = \begin{bmatrix} \frac{u(t_0,x_0)\cdot u_x(t_0,x_0)}{||f_2(t_0)\odot f_3(t_0)||} \\ \vdots \\ \frac{u(t_i,x_j)\cdot u_x(t_i,x_j)}{||f_2(t_i)\odot f_3(t_i)||} \\ \vdots \\ \frac{u(t_m,x_n)\cdot u_x(t_m,x_n)}{||f_2(t_m)\odot f_3(t_m)||} \end{bmatrix}; \tag{7}$$

The evolutionary part of the algorithm is aimed at the selection of a shortlist of the terms. In the shortlist, one of the terms is randomly selected as the target. The target term is described with the weighted combination of the other terms in the list. In the beginning, a randomized collection of possible solutions, which is called population, is declared. Each of these individuals represents one set of terms with the selected target, that can be interpreted as the right part of the equation, and features, a linear combination of which composes the left part. In order to perform selection, the fitness function is introduced in Eq. 8 as the inverse value of L_2-norm of differences between target $\mathbf{F}_{target}$ and selected combination of features $\mathbf{F}$ with weighs α, obtained by the sparse regression. Therefore, the task of the evolutionary algorithm can be reduced to the obtainment of the equation structure with the highest value of fitness-function.

$$f_{fitness} = \frac{1}{\|\mathbf{F}\cdot\alpha - \mathbf{F}_{target}\|_2} \longrightarrow max \tag{8}$$

The genotype of the individual is represented by the composition of the encoded structures for each term. These encodings contain powers of each token in the term. The evolution of individuals is performed both by mutation and by a crossover in every iteration step. The mutation for an individual is introduced as the random change addition, deletion of alteration of factors in its terms. For example, this can result in shift of equation term u_t to $u_{xx} * u_t$ or $u_x * u_{tt}$ to $u_x * u_t$. The elitism that helps to preserve the best-detected candidate consists of the preservation of some population individuals with the highest fitness function value during the mutation step.

Crossover is the part of the evolutionary mechanism of gene exchange between two individuals in order to produce offsprings that can have higher fitness values. In the task of data-driven equation derivation, it can be introduced by the exchange of terms between equations. In order to produce units with higher fitness values, the crossover must be held between selected individuals. Various tests have proved that the fastest convergence to the desired solution can be achieved with the tournament selection. In this policy, several tournaments, where the unit with the highest fitness value is selected for a further crossover, is held between individuals of the population. After that, parents for the offsprings are randomly chosen between the tournament winners. In contrast to the simple selection of several individuals with the highest fitness function values for reproduction, this approach can let the offsprings take good qualities for less-valuable individuals of the population.

The next essential element of the proposed data-driven algorithm is sparse regression. Its main application is the detection of the equation structure among the set of possible terms. With no original information about the equation structure and the correct number of terms in it, that is more secure to allow the equation to have a higher number of possible term candidates. Therefore some form of filtration has to take place. The main instrument in this phase is the Least Absolute Shrinkage and Selection Operator (LASSO). In contrast to other types of regression, LASSO can reduce the number of non-zero elements of the weights vector, giving zero coefficients to the features, that are not influential to the target.

The minimized functional of the LASSO regression (Eq. 9) takes the form of the sum of two terms. First is the squared error between vectors of target, denoted as $\mathbf{F_{target}}$, and vector of predictions, obtained as the dot product of matrix of features $\mathbf{F}$ and vector of weights α, while second in the L_1-norm of the weights vector, taken with sparsity constant λ:

$$||\mathbf{F}\alpha - \mathbf{F}_{target}||_2^2 + \lambda||\alpha||_1 \rightarrow \min_{\alpha} \tag{9}$$

The main imperfection of the LASSO regression is its disability to acquire actual values of the coefficients. To obtain the actual coefficients of the resulting PDE, final linear regression over discovered influential terms is performed. In the final step, non-zero weights from the LASSO are rescaled with original unnormalized data as features and the target.

The pseudo-code for the resulting algorithm is provided in App. A

4 Numerical experiments

4.1 Synthetic data

The analysis of the algorithm performance, just as in the case of one space dimension or on the clean data, first of all, shall be held on the synthetic data. This simplification can show the response of the result to various types and magnitude of noise, which is generally unknown on the measurement data. As in the previous studies, the solution of the wave equation with two spatial variables Eq. 10, where t - time , x, y - spatial coordinates, u - studied function (for example, vertical displacement of membrane), and $\alpha_1 = \alpha_1 = 1$ was taken as the synthetic data. The algorithm has proved to be able to detect the correct structure of the equation with the clean data, while on the noisy data, additional terms or completely wrong structures have been detected.

$$\frac{\partial^2 u}{\partial t^2} = \alpha_1 \frac{\partial^2 u}{\partial x^2} + \alpha_2 \frac{\partial^2 u}{\partial y^2} \tag{10}$$

Several noise addition experiments were held on the synthetic data: first of all, in a fraction of points (40% of total number) the noise of various magnitudes have been added: ($\mu = 0; \sigma = n * ||u(t)||,\ n = \overline{0.1, 0.8}$). After that, the algorithm has been applied to this data. The results of the experiment are as follows:

the method is successfully able to detect the structure of the equation for the interval of noise levels up to 14.9 %, which corresponds to the standard deviation of Gaussian noise in the interval [0, 0.35], multiplied by a norm of the field in the time frame. The weights errors in this interval are minor, as it is shown in the Tab. 2. With higher noise levels (in the interval between 14.9% and 15.67%), the algorithm detects additional terms that are not present in the original equation, which may result both in the distortion of equation structure and incorrect weights calculation. Finally, with high noise levels, the proposed algorithm can lose grasp of the correct structure of the equation.

Table 2. Discovered structures of the equations for the specific noise levels

Noise level of input data (%)	equation
0	$\frac{\partial^2 u}{\partial t^2} = 1.00\frac{\partial^2 u}{\partial x^2} + 1.00\frac{\partial^2 u}{\partial y^2}$
8.3	$\frac{\partial^2 u}{\partial t^2} = 1.02\frac{\partial^2 u}{\partial x^2} + 1.01\frac{\partial^2 u}{\partial y^2}$
10.9	$\frac{\partial^2 u}{\partial t^2} = 1.04\frac{\partial^2 u}{\partial x^2} + 0.99\frac{\partial^2 u}{\partial y^2}$
13.1	$\frac{\partial^2 u}{\partial t^2} = 0.96\frac{\partial^2 u}{\partial x^2} + 0.99\frac{\partial^2 u}{\partial y^2}$
14.9	$\frac{\partial^2 u}{\partial t^2} = 0.95\frac{\partial^2 u}{\partial x^2} + 1.2\frac{\partial^2 u}{\partial y^2}$
15.67	$\frac{\partial^2 u}{\partial t^2} = 0.84\frac{\partial^2 u}{\partial x^2} + 0.63\frac{\partial^2 u}{\partial y^2} + 0.12\frac{\partial u}{\partial y}$
16.45	$\frac{\partial^2 u}{\partial x^2}\frac{\partial^2 u}{\partial y^2} = 0$
17.88	$\frac{\partial^2 u}{\partial x^2}\frac{\partial^2 u}{\partial y^2} = 0$

In Fig. 3, the influence of the noise level added to the measured field on the derivative fields is shown.

In the other experiment with the same data set, the noise of relatively high magnitudes was added to a minor fraction of points (5% of the total number). In this case, the framework has shown similar results to the previous experiment: until the noise level of approximately 15%, the discovered structure was correct. On the data with higher noise magnitudes, the errors in the structure of the equation occurred. This experiment has shown that in the studied cases, the main limiting factor for the performance of the algorithm with implemented preprocessing for noise reduction is the noise level and not the distribution of noise across the studied field.

4.2 Physical measurements data

For the validation of the model, the dynamics of the two-dimensional field of sea surface height (SSH) data from the NEMO ocean model for the Arctic region for a modelling month with the resolution of an hour has been used. The part of the area in the center of the Barents Sea was selected for the numerical experiments. The area is known to have strong tides, which can lead to the discovery of the time-dependent equation. It is necessary to emphasise that despite existing

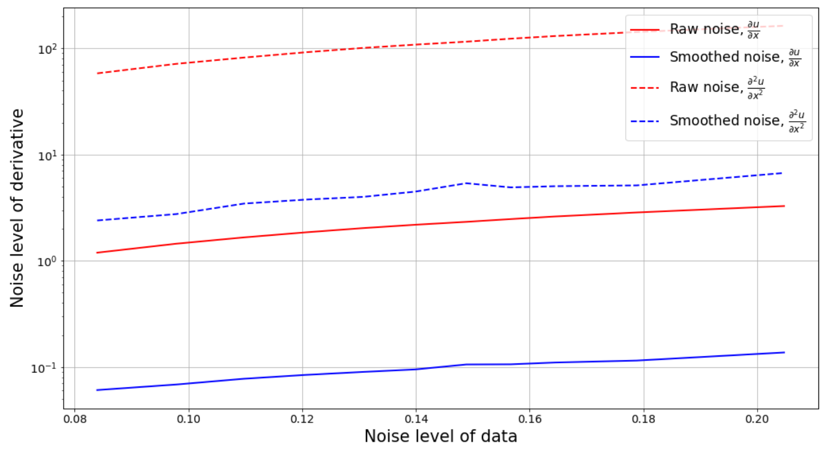

Fig. 3. Noise levels of calculated first and second (dashed line) time derivatives, related to noise levels of input data

Tidal equations, there is no single analytical equation for the specific case of the dynamics of the SSH in this region due to the overlapping of processes of different natures.

After the application of the framework to the data, the structure of the equation in form Eq. 11 has been acquired.

$$\frac{\partial u}{\partial x} = -0.0506\frac{\partial u}{\partial t} - 0.0053\frac{\partial^2 u}{\partial t^2} \tag{11}$$

To validate the result of the algorithm, Eq. 11 was solved, and the calculated field was compared with the initial one, as shown in Fig. 4. Since there is second-order time derivative and first spatial derivative, the initial conditions (first two time frames to represent the field and its first time derivative for the beginning of studied period) and the boundary condition on one edge of the studied area are set. The graphs of daily sea surface height dynamics from reanalysis and equation solution is presented in Fig. 5. The metrics of quality show that the discovered equation can describe the equation well: $RMSE = 0.0434, MAE = 0.0446$ for the field with values in interval between approximately 0.5 and 0.9.

4.3 Comparison with other methods

The experiments with conditions, similar to the one in [4], have been performed in order to compare the proposed algorithm with existing state-of-the-art methods. Due to limitations of the framework, namely the ability to derive only a single equation, instead of the system Eq. 12, that are utilized in the referenced article, a test was performed on similar Eq. 13. Additional difficulties to the comparison were contributed by unknown initial and boundary conditions in the referenced experiment.

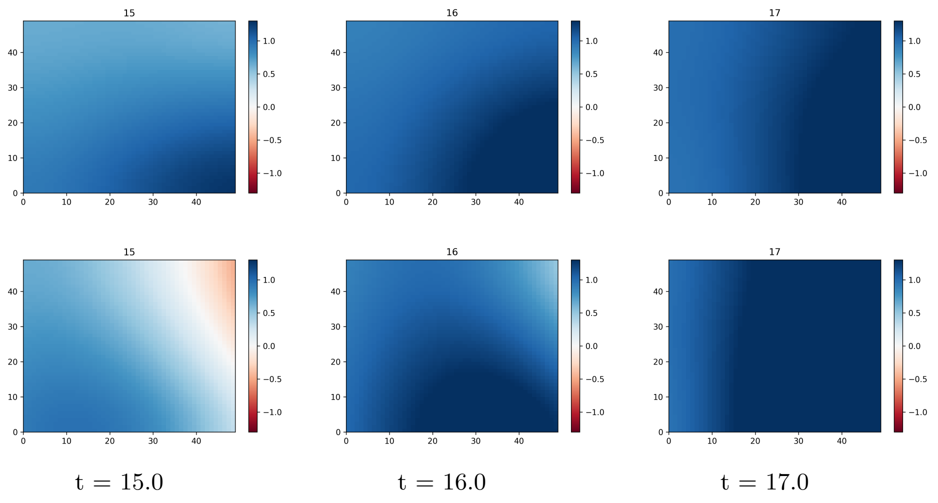

Fig. 4. Example of SSH field, obtained from reanalysis (upper row) and the same field from Eq. 11, (lower row) for 3 time frames

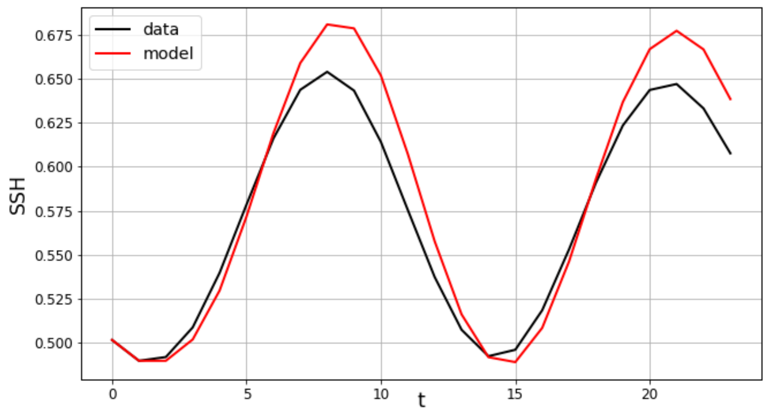

Fig. 5. Dynamics of sea surface height for September 18, 2013: reanalysis (denoted as data) and solution of the equation, obtained from framework, denoted as model, for the center of the studied area

$$\begin{cases} \frac{\partial U}{\partial t} = -U\nabla U + \nu \Delta U, U = (u,v)^T \\ U|_{t=0} = U_0(x,y) \end{cases} \tag{12}$$

$$\frac{\partial u}{\partial t} = (\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) + u(\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y}) \tag{13}$$

Eq. 13 was solved using finite differences, and the noise from normal distribution was added to simulate the aforementioned experiment. The preprocessing phase, described in detail in previous sections and involving smoothing and derivatives calculation, was performed to reduce the influence of noise on the resulting equation.

The added noise was created from $k \times \max_{x,y,t} u(x, y, t) \times N(0, 1)$. The experiments have been conducted with the value of k, equals to 0.001, as in the compared study [4], and the discovered equations had correct structures. As the framework output, we will consider the closest to the correct structure of the equation, obtained on the grid of sparsity constant values.

Table 3. Discovered structures of the equations for the specific noise levels

k	Noise level of input data (%)	equation
0.0005	0.25%	$\frac{\partial u}{\partial t} = (0.999\frac{\partial^2 u}{\partial x^2} + 1.000\frac{\partial^2 u}{\partial y^2})+$ $+u(1.001\frac{\partial u}{\partial x} + 1.000\frac{\partial u}{\partial y})$
0.00075	0.49%	$\frac{\partial u}{\partial t} = (1.001\frac{\partial^2 u}{\partial x^2} + 1.001\frac{\partial^2 u}{\partial y^2})+$ $+u(0.999\frac{\partial u}{\partial x} + 0.999\frac{\partial u}{\partial y})$
0.001	0.97%	$\frac{\partial u}{\partial t} = (1.000\frac{\partial^2 u}{\partial x^2} + 0.999\frac{\partial^2 u}{\partial y^2})+$ $+u(1.000\frac{\partial u}{\partial x} + 1.000\frac{\partial u}{\partial y})$
0.00125	4.2%	$\frac{\partial u}{\partial t} = (1.001\frac{\partial^2 u}{\partial x^2} + 1.002\frac{\partial^2 u}{\partial y^2})+$ $+u(0.998\frac{\partial u}{\partial x} + 0.999\frac{\partial u}{\partial y})$
0.0015	6.3%	$\frac{\partial u}{\partial t} = (0.996\frac{\partial^2 u}{\partial x^2} + 0.998\frac{\partial^2 u}{\partial y^2})+$ $+u(1.000\frac{\partial u}{\partial x} + 1.003\frac{\partial u}{\partial y}) - 0.0034\frac{\partial^2 u}{\partial y^2}\frac{\partial u}{\partial x}$

In these tests, shown noise resistance corresponds with other framework applications and is somewhat better than in the experiment in the compared experiment. Despite the insignificant difference in the coefficient k (0.001 versus 0.00015), the noise level difference is significant (1% versus 6.3%). For the noise levels approximately below 5% the correct equations were detected. After that, the structure of the equation deteriorates, which manifests in wrong weights and the presence of additional terms/lack of mandatory ones.

5 Conclusion

The proposed method has proven to be suitable for the data-driven derivation of equations, that can be used for modelling of various physical processes. The robustness of the algorithm to the noise in the input data that is provided by improved preprocessing of data, which included the application of kernel smoothing to the input data matrices, and derivative calculation by differentiating polynomials, fit over points inside a selected window, allows the framework applicable to the real-world problems. Even in the cases of substantial noise in the input data, the resulting equations had the correct structures and, therefore, can correctly describe the studied system. Other notable points about the algorithm operation can be stated:

– To achieve a good quality of the resulting processes, the areas, localizing different processes, should be separated and studied on their own. The example of such approach to the problem was presented in the case of real-world data processing when the area that is already known to have strong time dependencies of sea surface height was separated, and the equation for it was derived;
– The meta-parameters of the algorithm have a strong influence on the final result. For example, low values of sparsity constant can lead to the presence of additional terms in the equation, while its higher than optimal values can completely distort the equation structure. Therefore, mechanisms of meta-parameter selection should be implemented in the further development of the method.

Areas of the further development of the framework can include the derivation of a more generalized class of equations, using similar techniques, not limiting the results in a class of partial differential equations. Additionally, the equations for vector variables or even systems of equations can be the next targets for the work.

Source code is publicity available at GitHub [6].

Acknowledgements

This research is financially supported by The Russian Scientific Foundation, Agreement #19-71-00150

References

1. Berg, J., Nyström, K.: Neural network augmented inverse problems for pdes (2018), https://arxiv.org/abs/1712.09685
2. Chang, H., Zhan, D.: Machine learning subsurface flow equations from data. Computational Geosciences (2018). https://doi.org/10.1007/s10596-019-09847-2
3. Knowles, I., Le, T., Yan, A.: On the recovery of multiple flow parameters from transient head data. Journal of Computational and Applied Mathematics **169**(1), 1–15 (2004)
4. Long, Z., Lu, Y., Ma, X., Dong, B.: Pde-net: Learning pdes from data (2017), https://arxiv.org/abs/1710.09668
5. Maslyaev, M., Hvatov, A., Kalyuzhnaya, A.: Data-driven partial derivative equations discovery with evolutionary approach. In: International Conference on Computational Science. pp. 635–641. Springer (2019)
6. NSS Team: Fedot E* algotirhms. https://github.com/ITMO-NSS-team/FEDOT.Algs (2020)
7. Piche, R.: Automatic numerical differentiation by maximum likelihood estimation of state-space model (2016), https://arxiv.org/abs/1610.04397v1

8. Qin, T., Wu, K., Xiu, D.: Data driven governing equations approximation using deep neural networks (2018), https://arxiv.org/abs/1811.05537
9. Raissim, M.: Deep hidden physics models: Deep learning of nonlinear partial differential equations (2018), https://arxiv.org/abs/1801.06637
10. Rudy, S.H., Brunton, S.L., Proctor, J.L., Kutz, J.N.: Data-driven discovery of partial differential equations. Science Advances **3**(4), e1602614 (2017)
11. Schaeffer, H., Caflisch, R., Hauck, C.D., Osher, S.: Learning partial differential equations via data discovery and sparse optimization. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science (2017). https://doi.org/473(2197):20160446
12. Schaeffer, H.: Learning partial differential equations via data discovery and sparse optimization. Proc. R. Soc. A **473**(2197), 20160446 (2017)

A Pseudo-code of the algorithm

```
Input: set of elementary tokens $T$, symbolically representing constant, initial
       function, and its various derivatives; set of function measurements
       from the studied field
Parameters : $M$ - number of token combinations in a single individual; $k$ -
       number of elementary tokens in a combination; $n\_pop$ -
       number of candidate solutions in the population; evolutionary
       algorithm parameters: number of epochs $n_{epochs}$, mutation
       $r_{mutation}$ & crossover rates $r_{crossover}$, part of the population,
       allowed for procreation $a_{proc}$, number of individuals, refrained
       from mutation (elitism) $a_{elite}$; sparse regression parameter -
       sparsity constant $\lambda$
Result: The structure of the partial differential equation with
       thecorresponding weights, best fitting the input field
Apply smoothing to the measurements & calculate the derivatives; Generate
  population P of individuals, representing equation, of size $n\_pop$, with $M$ -
  random permutations of $k$ tokens to form sets $C^j$;
for epoch = 1 to $n_{epochs}$ do
    for individual in population do
        Apply sparse regression to individual to calculate weights;
        Calculate fitness function to individual;
    end
    Hold tournament selection and crossover;
    for individual in population except $n\_pop \times a_{elite}$ "elite" ones do
        Mutate individual;
    end
end
Select the individual with highest fitness function value as the final structure
  of the solution to the problem;
Calculate true weights of the equation, using linear regression.
```

Algorithm 1: The pseudo-code of the algorithm operation