Global Optimization, Ordinary Differential Equations and Infinity Computing

Marat S. Mukhametzhanov

DIMES, Universitá della Calabria
m.mukhametzhanov@dimes.unical.it

12 Febbraio 2020

Research projects

This work was supported by the following INdAM GNCS Projects:

- "L'ottimizzazione globale, equazioni differenziali ordinarie e l'Infinity Computing" ("Giovani ricercatori 2019")
- INdAM GNCS 2018 Research Project "Numerical methods in optimization and ODEs"

"Giovani Ricercatori" 2019

Main results obtained in the framework of the research project have been published in the following papers:

- A. Falcone, A. Garro, M.S. Mukhametzhanov, Ya.D. Sergeyev, "A Simulink-based Infinity Computer Simulator and some applications", Lecture Notes in Computer Science, vol. 11974, 2019, https://doi.org/10.1007/978-3-030-40616-5_31, in press.
- M.S. Mukhametzhanov, Ya.D. Sergeyev, "The Infinity Computer vs. Symbolic Computations: First Steps in Comparison", AIP Conference Proceedings, 2019, in press.
- M.S. Mukhametzhanov, R. Cavoretto, A. De Rossi, "An Experimental Study of Univariate Global Optimization Algorithms for Finding the Shape Parameter in Radial Basis Functions", Communications in Computer and Information Science, vol 1145, 2019, pp. 326–339.
- D.E. Kvasov, M.S. Mukhametzhanov, M.C. Nasso, Ya.D. Sergeyev, "On Acceleration of Derivative-Free Univariate Lipschitz Global Optimization Methods", Lecture Notes in Computer Science, vol. 11974, 2019, https://doi.org/10.1007/978-3-030-40616-5_38, in press.

Outline

Section 1

The Infinity Computer

The Infinity Computer

- Infinity Computer – a new type of a supercomputer allowing one to work numerically with finite, infinite, and infinitesimal numbers in the same way.
- It is described in the EU patent 1728149; RF Patent 2395111; USA patent 7,860,914.
- The numbers are represented using the positional numeral system with the infinite radix ① (called *grossone*) introduced as the number of elements of the set of natural numbers.
- All the computations within the Infinity Computing framework are *numerical* and not symbolical.
- The symbols ∞, ω, $\aleph_0, \aleph_1, \ldots$ are excluded from our language.

Positional system with infinite radix

- A number C is represented using the positional numeral system with the radix ①:

$$C = c_{p_m}①^{p_m} + \ldots + c_{p_1}①^{p_1} + c_{p_0}①^{p_0} + c_{p_{-1}}①^{p_{-1}} + \ldots + c_{p_{-k}}①^{p_{-k}}.$$

- Finite numerals c_i are called *grossdigits*. They can be both positive and negative and expressed using traditional computational frameworks.
- Numerals p_i are called *grosspowers*. They can be finite, infinite, and infinitesimal (Only finite grosspowers are considered in this work):

$$p_m > p_{m-1} > \ldots > p_1 > p_0 = 0 > p_{-1} > \ldots p_{-(k-1)} > p_{-k}.$$

- Numbers expressed in the new numeral system:
  - Finite numbers: $C = c_0①^0 = c_0$.
  - Infinite numbers: ①, $①^{1.7}$, $①^2$, $2.1①^{3.2} - 1.1①^{1.2}0.8①^0 1.9①^{-1}$, etc.
  - Infinitesimal numbers:$①^{-1} = \frac{1}{①}$, $①^{-2}$, $2①^{-2.1} - 0.9①^{-5.8}1.8①^{-9}$, etc.

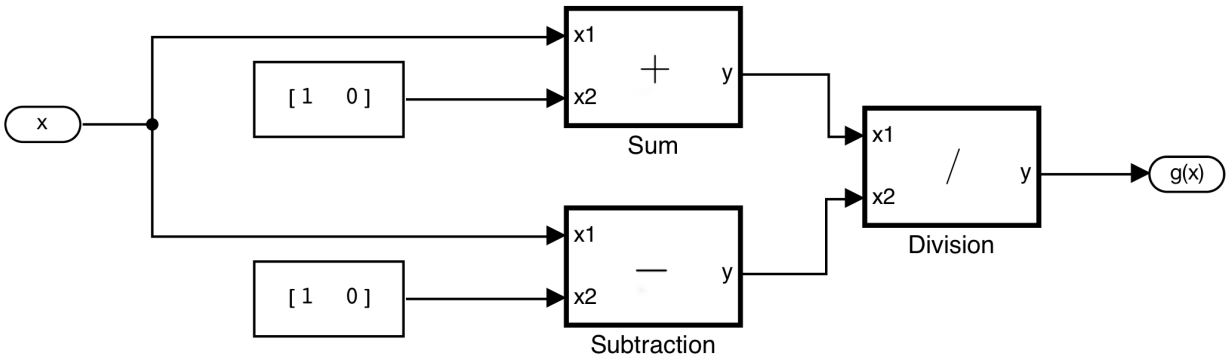
Simulink-based software solution to the Infinity Computer

- A Simulink-based software solution to the Infinity Computer has been proposed.
- The solution is general-purpose and domain-independent.
- It can be used in all industrial and scientific domains where a high level of accuracy in the calculations represents a key factor (e.g., Cyber-Physical Systems, Robotics and Automation);
- It allows engineers to focus on the specific aspects of their system, without dealing with low level APIs as well as complex procedures of the emulator of the *Infinity Computer Arithmetic C++ library (ICA-lib)* that can distract them from their high level design.

A. Falcone, A. Garro, M.S. Mukhametzhanov, Ya.D. Sergeyev, A Simulink-based Infinity Computer Simulator and some applications, Lecture Notes in Computer Science, vol. 11974, 2019, https://doi.org/10.1007/978-3-030-40616-5_31, in press.

An example of a univariate function

Simulink block of the function $g(x) = (x + 1①^0)/(x - 1①^0)$ using the Infinity Computer Arithmetic blocks of the elementary operations $+$, $-$ and $/$



In the Simulink-based solution, each number is represented by the variable sized matrix, where the first column contains the grossdigits and the second column contains the grosspowers. E.g., the number 1 is represented by the matrix $\begin{bmatrix}1 & 0\end{bmatrix}$, while the number $3①^0 1①^{-1}$ is represented by the matrix $\begin{bmatrix}3 & 0\\ 1 & -1\end{bmatrix}$.

Section 2

Ordinary Differential Equations and the Infinity Computer

New techniques for computing Lie derivatives of a function.

- MOTIVATION # 1: There is a certain numerical evidence that some symmetric multi-derivative methods for solving ODEs posses good properties, but the drawback of computing derivative limited the use of these methods;
- MOTIVATION # 2: Higher-order derivatives may be safely and efficiently computed with the aid of the Infinity Computer.

Exact derivatives using Infinity Computer

Theorem

Suppose that:
(i) for a function $g(x)$ evaluated by a procedure implemented on the Infinity Computer there exists an unknown Taylor expansion in a finite neighborhood δu of a finite point u;
(ii) $g(x), g'(x), g''(x), \cdots g^{(k)}(x)$ assume finite values or are equal to zero for $x \in \delta(u)$;
(iii) $g(x)$ has been evaluated at a point $u + ①^{-1} \in \delta(u)$. Then the Infinity Computer returns the result of this evaluation in the positional numeral system with the infinite radix ① in the following form
$g(u + ①^{-1}) = c_0①^0 c_1①^{-1} c_2①^{-2} \ldots c_{(k-1)}①^{-(k-1)} c_k①^{-k}$, where
$g(u) = c_0, g'(u) = c_1, g''(u) = 2!c_2, \cdots g^{(k)}(u) = k!c_k$

Y. D. Sergeyev, Higher order numerical differentiation on the Infinity Computer, Optimization Letters 5(4) (2011) 575–585.

Computation of Lie derivative

Useful when using multiderivatives numerical schemes to approximate the solution of initial value problems:

$$\begin{cases} y' = f(y(t)), & t \in [t_0, T], \\ y(t_0) = y_0, \end{cases}$$

f scalar and autonomous:

$$\begin{aligned} y'(t_0) &= D^0 f(t_0) = f(y_0), \\ y''(t_0) &= D^1 f(t_0) = f'(y_0) f(y_0), \\ y'''(t_0) &= D^2 f(t_0) = f''(y_0)(f(y_0))^2 + (f'(y_0))^2 f(y_0), \\ y^{(iv)}(t_0) &= D^3 f(t_0) = f'''(y_0)(f(y_0))^3 + 4 f''(y_0) f'(y_0)(f(y_0))^2 + (f'(y_0))^3 \end{aligned}$$

f not scalar: the analytical computation of the j-th derivative involves a tensor of order j.

Computing derivatives - First Algorithm - Strategy (a)

Perform k infinitesimal steps starting at time t_i using the explicit Euler formula with stepsize $①^{-1}$:

$$y_{i,1} = y_i + ①^{-1} f(y_i), \quad y_{i,2} = y_{i,1} + ①^{-1} f(y_{i,1}),$$

$$\ldots, \quad y_{i,k} = y_{i,k-1} + ①^{-1} f(y_{i,k-1}).$$

Compute the derivatives by means of the forward differences

$$F^k_{①^{-1}}[y_{i,0}, y_{i,1}, \ldots, y_{i,k}] = \sum_{j=0}^{k} (-1)^j \binom{k}{j} y_{i,k-j}, \quad y_{i,0} = y_i.$$

$$y^{(k)}(t_i) = \frac{F^k_{①^{-1}}[y_{i,0}, y_{i,1}, \ldots, y_{i,k}]}{①^{-k}} + O(①^{-1})$$

Ya.D. Sergeyev, *Solving ordinary differential equations by working with infinitesimals numerically on the Infinity Computer*, Applied Mathematics and Computation, 219(22), pp. 10668–10681,(2013)

Computing derivatives - Second Algorithm - Strategy (b)

Finite differences are employed directly on the value of f as follows:

$$y^{(k)}(t_i) = D^{k-1} f(y_i) = \frac{F_{①^{-1}}^{k-1}[f(y_{i,0}), f(y_{i,1}), \ldots, f(y_{i,k-1})]}{①^{-(k-1)}} + O(①^{-1}).$$

- Strategy (a) and (b) are equivalent.
- Stretegy (a) requires grossdigits corresponding to the exponents of grossone from 0 to $-k$.
- Stretegy (b) requires grossdigits corresponding to the exponents of grossone from 0 to $-k-1$.
- Strategy (b) is more efficient

Iavernaro, F.; Mazzia, F.; Mukhametzhanov, M.; Sergeyev, Y. Conjugate symplecticity properties of Euler–Maclaurin methods and their implementation on the Infinity Computer. 2018, arXiv:1807.10952, *Appl. Numerical Mathem.* in press.

Computing derivatives - Third Algorithm - Strategy (c)

Based on the Explicit Taylor method for approximating the differential equations

Theorem

Let us suppose that for the solution $y(t)$ of the ordinary differential equation $y' = f(t, y)$ it is known the value $y_i = y(t_i)$ at the point t_i and all the derivative $y_i^{(j)} = y^{(j)}(t_i), j = 1, ..., k$ and that they assume finite values or are zero let $y_{i,k} = y_i + ①^{-1} y_i' + \frac{①^{-2}}{2} y_i'' + \cdots + \frac{①^{-k}}{k!} y_i^{(k)}$ the truncated Taylor expansion of $y(t_i + ①^{-1})$.
Then the coefficient of order k of $①^{-1}$ of $f(y_{i,k})$ multiplied by $k!$ is equal to the exact derivative $y_i^{(k+1)} = D^k f(y(t_i))$.

Computing derivatives - Third Algorithm - Strategy (c)

The algorithm to obtain the derivative up to order k is the following:
starting from $f(y(t_i))$, for $j = 1, ..., k-1$,
compute the Taylor approximation of $y(t_i + ①^{-1})$ of order j,

$$y_{i,j} = y_i + \sum_{i=1}^{j} \frac{①^{-j}}{j!} y_i^{(j)},$$

compute $f(y_{i,j})$ using grosspower up to $-j$.
The coefficient of the grosspower $-j$ multiplied by $(j)!$ is equal to:

$$y_i^{(j+1)} = D^j f(y(t_i)).$$

Computing derivatives - Example

Computation of the first 3 derivatives $y'(t_0)$, $y''(t_0)$, and $y'''(t_0)$ of the solution $y(t)$ at the point $t_0 = 0$ of :

$$\frac{dy}{dt} = \frac{y - 2ty^2}{1 + t}, \; y(t_0) = 0.4,$$

exact solution:

$$y(t) = \frac{1 + t}{2.5 + t^2}.$$

Exact values:

$$y'(t_0) = 0.4, \quad y''(t_0) = -0.32, \quad y'''(t_0) = -0.96.$$

Computing derivatives - Example, strategy (c).

Perform 2 steps of Taylor methods, starting at y_0 and increasing the order of approximation:

$$
\begin{aligned}
y_{1,0} &= y_0 \\
f(t_0, y_0) &= 0.4 \qquad \rightarrow y_0' = 0.4 \\
y_{1,1} &= y_0 + f(t_0, y_0)①^{-1} = 0.4 + 0.4①^{-1} \\
f(t_0 + ①^{-1}, y_{1,1}) &= 0.4 - 0.32①^{-1} \qquad \rightarrow y_0'' = -0.32 \\
y_{1,2} &= y_0 + f(t_0, y_0)①^{-1} + \frac{1}{2} y_0''①^{-2} \\
&= 0.4 + 0.4①^{-1} - 0.16①^{-2} \\
f(t_0 + ①^{-1}, y_{1,2}) &= 0.4 - 0.32①^{-1} - \frac{0.96}{2}①^{-2} \qquad \rightarrow y_0''' = -0.96.
\end{aligned}
$$

We have computed y_0'' using grosspowers up to -1 and y_0''' using grosspowers up to -2.

The Infinity Computer vs. Symbolic Computations: evaluation times

- The first k, $k = 0, 1, ..., 10$, derivatives have been calculated for the first 10 Pintér's univariate test functions:

$$f_s(x) = 0.025(x - x_s^*)^2 + \sin^2[(x - x_s^*) + (x - x_s^*)^2] + \sin^2(x - x_s^*), \ s = 1, ..., 10.$$

- First, symbolic derivatives have been calculated and the execution time for their generation has been estimated.
- Each derivative has been evaluated 100 times on the uniform grid

$$x_i, \ i = 1, 2, ..., 100, \ x_1 = -5, \ x_{100} = 5.$$

- The average execution time over 100 evaluations has been then calculated.
- In order to calculate the respective $k-$th derivatives, the Infinity Computer has evaluated each test function 100 times at the points

$$x_i + ①^{-1}.$$

  The average execution time for each test function has been calculated.
- Each derivative for both the symbolic computational framework and the Infinity Computer has been first evaluated at the point $x_0 = -5$ before the evaluation at the grid, since in MATLAB the first evaluation is usually slower than the others.
- Values of the derivatives obtained by the Infinity Computer and by the symbolic differentiation at each evaluation point for each test function coincided up to machine precision.

The Infinity Computer vs. Symbolic Computations: evaluation times

k	T_k^{gen}	T_k^{sym}	T_k^{IC}	Speedup
0	3.34E-02	1.64E-03	7.79E-05	20.98
1	3.68E-02	3.42E-03	2.27E-04	15.05
2	4.00E-02	5.41E-03	3.98E-04	13.59
3	4.35E-02	7.37E-03	6.84E-04	10.78
4	4.70E-02	9.41E-03	7.51E-04	12.52
5	5.04E-02	1.15E-02	1.15E-03	10.05
6	5.39E-02	1.37E-02	1.23E-03	11.16
7	5.75E-02	1.60E-02	1.75E-03	9.15
8	6.12E-02	1.84E-02	1.87E-03	9.86
9	6.51E-02	2.09E-02	2.49E-03	8.42
10	6.89E-02	2.37E-02	2.64E-03	8.97

Average execution times of computation of the first k, $k = 0, 1, ..., 10$, derivatives of the Pintér's functions.

M.S. Mukhametzhanov, Ya.D. Sergeyev The Infinity Computer vs. Symbolic Computations: First Steps in Comparison, AIP Conf. Proceedings, 2019, in press.

Section 3

Univariate Global Optimization and the Infinity Computing

Univariate Lipschitz global optimization

Let us consider the following problem:

$$f^* = f(x^*) = \min_{x \in [a,b]} f(x), \quad D \subset \mathbb{R},$$

and

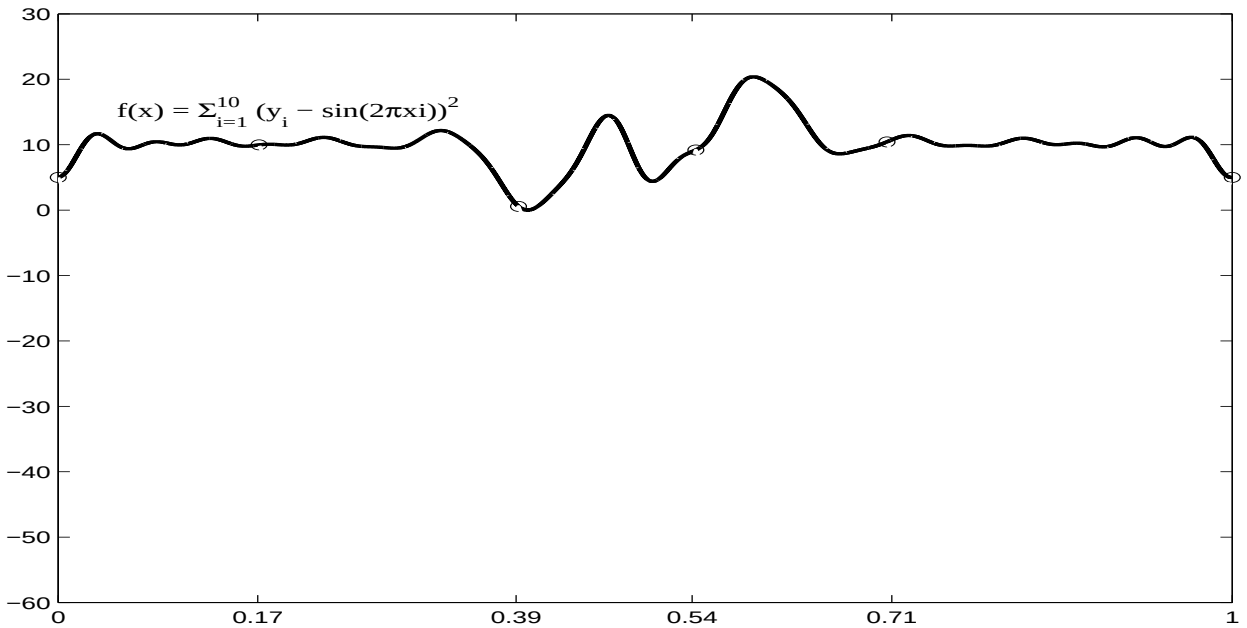
$$|f(x_1) - f(x_2)| \leq L|x_1 - x_2|, \ x_1, x_2 \in [a, b],$$

where L is the Lipschitz Constant, L is finite.

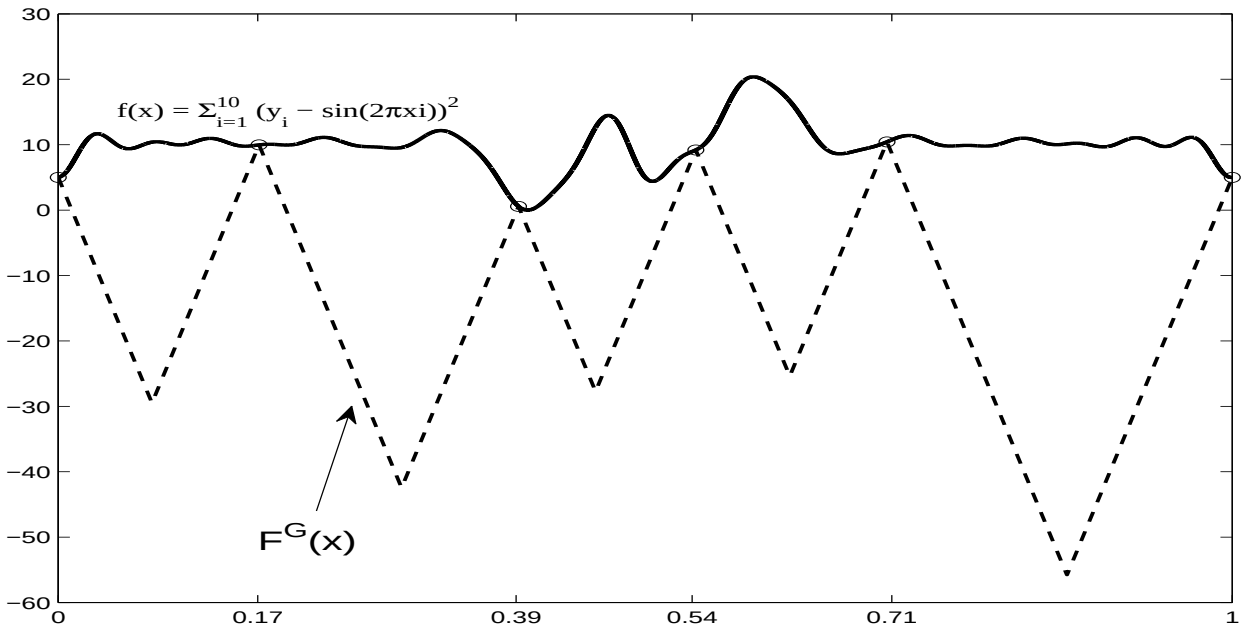
Objective function $f(x)$ is:

- Multiextremal;
- Non-differentiable;
- "Black-box";
- Hard to evaluate.

Univariate Lipschitz function



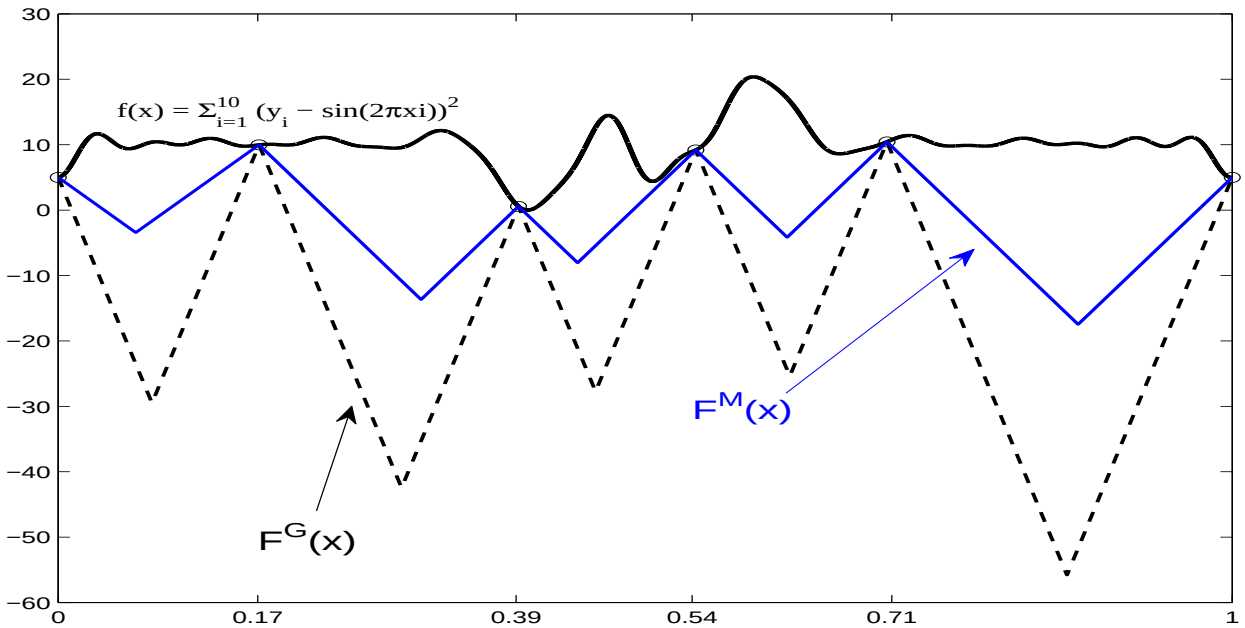
Global estimate of the Lipschitz constant



$F^G(x)$ – Using global estimate of the Lipschitz constant L.

.

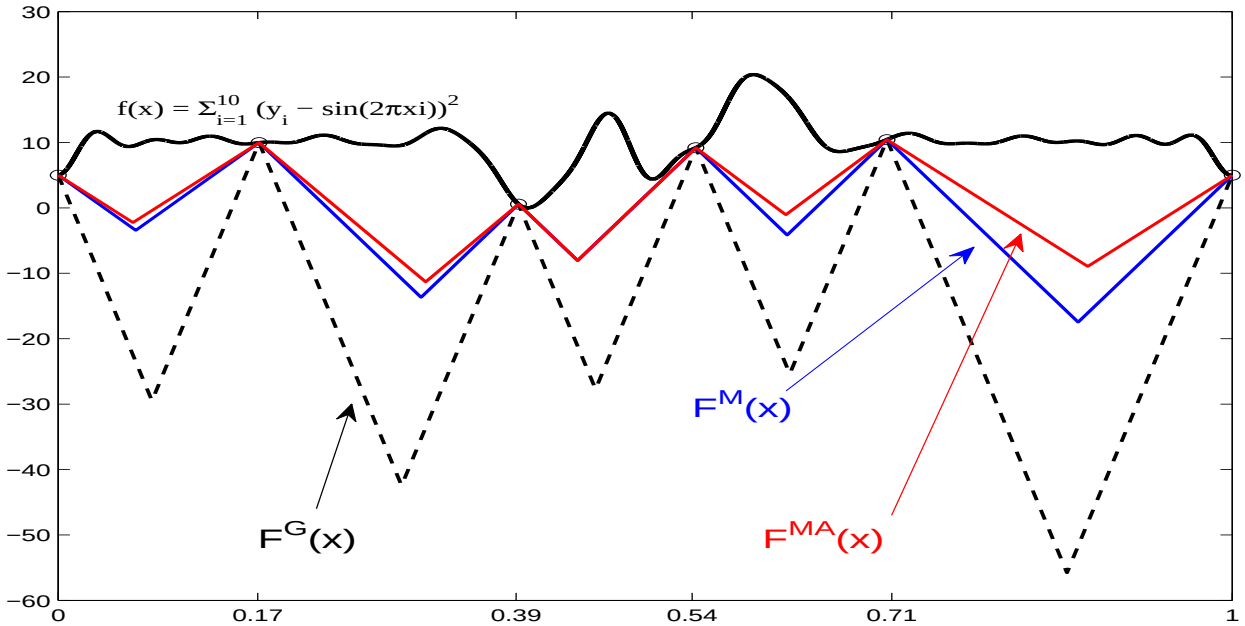
Global estimate vs. local tuning



$F^G(x)$ – Using global estimate of the Lipschitz constant L.

$F^M(x)$ – Using "Maximum" local tuning.

Another local tuning technique



$F^G(x)$ – Using global estimate of the Lipschitz constant L.

$F^M(x)$ – Using "Maximum" local tuning.

$F^{MA}(x)$ – Using "Maximum-Additive" local tuning.

Ya.D. Sergeyev, M.S. Mukhametzhanov, D.E. Kvasov, D. Lera The Derivative-free local tuning and local improvement techniques embedded in the univariate global optimization, Journal of Optimization Theory and Applications, 171(1):186–208, 2016.

Global optimization on the Infinity Computer

- The univariate Lipschitz global optimization algorithms have been implemented on the Infinity Computer.
- The methods work correctly even on scaled test functions $\alpha f(x) + \beta$, where α and β can be infinite or infinitesimal.
- In this case, the Lipschitz constant of the scaled function can be infinite or infinitesimal.

Ya. Sergeyev, D. Kvasov, M. Mukhametzhanov, On strong homogeneity of a class of global optimization algorithms working with infinite and infinitesimal scales, Communications in Nonlinear Science and Numerical Simulation, 59:319–330, 2018.

Local Tuning Technique for "Information" methods

A new Local Tuning technique has been proposed for univariate "Information" global optimization algorithms.

D.E. Kvasov, M.S. Mukhametzhanov, M.C. Nasso, Ya.D. Sergeyev On acceleration of derivative-free Lipschitz global optimization Methods, Lecture Notes in Computer Science, vol. 11974, 2019, https://doi.org/10.1007/978-3-030-40616-5_38, in press.

Local Tuning Technique for "Information" methods

The proposed local tuning technique has shown a very promising behavior with respect to existent local tuning methods. Results of preliminary experiments on two classes 100 of univariate test functions are presented:

Method	Class 1: Shekel		Class 2: Opposite Shekel	
	# $Trials_{avg}$	Success	# $Trials_{avg}$	Success
INF-LTIMO	54.48	91%	51.70	100%
INF-LTIMAO	61.34	95%	56.78	100%
INF-LTMIO	51.78	90%	50.66	100%

D.E. Kvasov, M.S. Mukhametzhanov, M.C. Nasso, Ya.D. Sergeyev On Acceleration of Derivative-Free Univariate Lipschitz Global Optimization Methods, Lecture Notes in Computer Science, vol. 11974, 2019, https://doi.org/10.1007/978-3-030-40616-5_38, in press.

An application: RBF interpolation

- Let the data set $X_n = \{x_1, ..., x_n\}$, $x_i \in \mathbb{R}^s$, $x_i \neq x_j$, $i, j = 1, ..., n$, and $F_n = \{f(x_1), \ldots, f(x_n)\}$ be given.
- Let $\mathcal{I}_f : \mathbb{R}^s \to \mathbb{R}$ be the radial basis function (RBF) interpolant of the form

$$\mathcal{I}_f(x) = \sum_{i=1}^{n} c_i \phi_\varepsilon(||x - x_i||_2), \quad x \in \mathbb{R}^s,$$

  where c_i are found from the interpolation conditions $\mathcal{I}_f(x_i) = f(x_i)$, $i = 1, ..., n$, and $\phi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a strictly positive definite RBF depending on a *shape parameter* $\varepsilon > 0$.
- Interpolation error Er can be calculated in different ways (e.g., as RMSE on test set, using cross validation, etc.)
- For different ε on the same data set and the same RBF, different interpolation error can be obtained.
- It is required to find the value ε^* and the value Er^* such that

$$Er^* = Er(\varepsilon^*) = \min Er(\varepsilon), \ \varepsilon \in [0, \varepsilon_{max}],$$

  where ε_{max} is large enough.

Global optimization problem related to RBF interpolation

- Performance of the RBF interpolation depends on the values of ε.
- Traditionally, "good" values of ε are calculated using exhaustive grid methods, local optimization schemes or even ignored.
- New univariate global optimization methods for solving the above mentioned optimization problem in RBF interpolation have been proposed in [1, 2].
- These methods use the univariate Lipschitz global optimization algorithms from [3, 4] as basic methods.


[1] R. Cavoretto, A. De Rossi, M.S. Mukhametzhanov, Ya.D. Sergeyev On the search of the shape parameter in radial basis functions using univariate global optimization methods, Journal of Global Optimization, 2019, in Press.

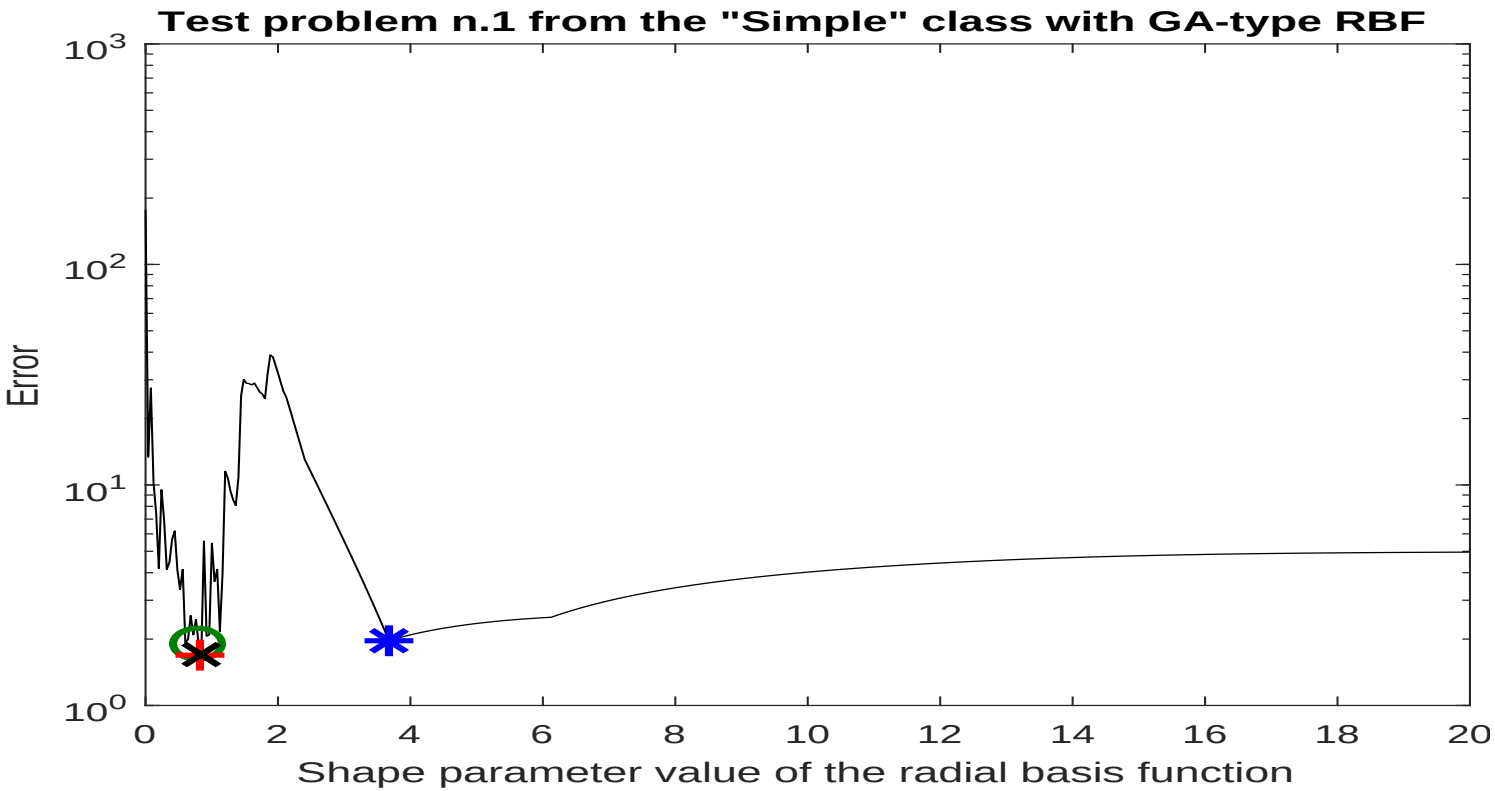

[2] M.S. Mukhametzhanov, R. Cavoretto, A. De Rossi An Experimental Study of Univariate Global Optimization Algorithms for Finding the Shape Parameter in Radial Basis Functions, Communications in Computer and Information Science, vol. 1145, 2019, pp. 326–339.


[3] Ya.D. Sergeyev, M.S. Mukhametzhanov, D.E. Kvasov, D. Lera The Derivative-free local tuning and local improvement techniques embedded in the univariate global optimization, Journal of Optimization Theory and Applications, 171(1):186–208, 2016.


[4] D.E. Kvasov, M.S. Mukhametzhanov, M.C. Nasso, Ya.D. Sergeyev On Acceleration of Derivative-Free Univariate Lipschitz Global Optimization Methods, Lecture Notes in Computer Science, vol. 11974, 2019,

Example of the optimized error function using Gauss-type RBF and GKLS-type test function

The best found values by the fixed grid methods with 100 and 500 nodes, respectively, are indicated as green “o” and black “x”, the local optimization method as blue “*”, and proposed global optimization method as red “+”.



Results obtained during the project

- New efficient differentiation techniques have been proposed for the Infinity Computer. These techniques are simple, fast and allow one to calculate higher order Lie derivatives without necessity of working with higher order tensors.
- It has been shown that the Infinity Computer allows one to calculate the exact solutions for different real-life problems with the accuracy of symbolic computations, but much faster, than symbolic computations do.
- A new Simulink-based software solution to the Infinity Computer has been proposed. This solution is general-purpose, domain-independent, and allows one to use all the potentiality of the Infinity Computer in Simulink without referring to the low-level implementation of the Infinity Computer arithmetic.
- A new acceleration technique has been proposed in the field of univariate Lipschitz global optimization. This technique allows one to find a "good" solution to the optimization problem in case of limited budget or hard to evaluate objective function.
- New Lipschitz global optimization algorithms have been proposed for solving real-life optimization problems in RBF interpolation.

The main obtained results have been published in

- A. Falcone, A. Garro, M.S. Mukhametzhanov, Ya.D. Sergeyev, "A Simulink-based Infinity Computer Simulator and some applications", Lecture Notes in Computer Science, vol. 11974, 2019, https://doi.org/10.1007/978-3-030-40616-5_31, in press.
- M.S. Mukhametzhanov, Ya.D. Sergeyev, "The Infinity Computer vs. Symbolic Computations: First Steps in Comparison", AIP Conference Proceedings, 2019, in press.
- M.S. Mukhametzhanov, R. Cavoretto, A. De Rossi, "An Experimental Study of Univariate Global Optimization Algorithms for Finding the Shape Parameter in Radial Basis Functions", Communications in Computer and Information Science, vol 1145, 2019, pp. 326–339.
- D.E. Kvasov, M.S. Mukhametzhanov, M.C. Nasso, Ya.D. Sergeyev, "On Acceleration of Derivative-Free Univariate Lipschitz Global Optimization Methods", Lecture Notes in Computer Science, vol. 11974, 2019, https://doi.org/10.1007/978-3-030-40616-5_38, in press.

Related publications

1. Ya. Sergeyev, D. Kvasov, M. Mukhametzhanov,
Scientific Reports, Nature Publishing Group, 8(453), 2018 (Q1).
2. F. Iavernaro, F. Mazzia, M. Mukhametzhanov, Ya. Sergeyev,
Applied Numerical Mathematics, 2019, in Press (Q2).
3. R. Cavoretto, A. De Rossi, M. Mukhametzhanov, Ya. Sergeyev,
Journal of Global Optimization, 2019, in press (Q1).
4. M. Gaudioso, G. Giallombardo, M. Mukhametzhanov,
Applied Mathematics and Computation, 318:312–320, 2018 (Q1).
5. Ya. Sergeyev, D. Kvasov, M. Mukhametzhanov,
Commun. in Nonlinear Science and Numerical Simulation, 59:319–330, 2018 (Q1).
6. D. Kvasov, M. Mukhametzhanov,
Applied Mathematics and Computation, 318:245–259, 2018 (Q1).
7. Ya. Sergeyev, D. Kvasov, M. Mukhametzhanov,
Mathematics and Computers in Simulation, 141:96 – 109, 2017 (Q1).
8. P. Amodio, F. Iavernaro, F. Mazzia, M. Mukhametzhanov, Ya. Sergeyev,
Mathematics and Computers in Simulation, 141:24 – 39, 2017 (Q1).
9. Ya. Sergeyev, M. Mukhametzhanov, D. Kvasov and D. Lera,
Journal of Optimization Theory and Applications, 171(1):186 - 208, 2016 (Q1).
10. Ya. Sergeyev, M. Mukhametzhanov, F. Mazzia, F. Iavernaro and P. Amodio,
International Journal of Unconventional Computing, 12(1):3-23,2016 (Q3).