DISCRETE AND CONTINUOUS MODELS AND APPLIED COMPUTATIONAL SCIENCE

Volume 31 Number 1 (2023)

Founded in 1993
Founder: PEOPLES' FRIENDSHIP UNIVERSITY OF RUSSIA

DOI: 10.22363/2658-4670-2023-31-1

Edition registered by the Federal Service for Supervision of Communications, Information Technology and Mass Media
Registration Certificate: ПИ № ФС 77-76317, 19.07.2019

ISSN 2658-7149 (online); 2658-4670 (print)
4 issues per year.
Language: English.

Publisher: Peoples' Friendship University of Russia (RUDN University).
Indexed by Ulrich's Periodicals Directory (http://www.ulrichsweb.com), Directory of Open Access Journals (DOAJ) (https://doaj.org/), Russian Index of Science Citation (https://elibrary.ru), EBSCOhost (https://www.ebsco.com), CyberLeninka (https://cyberleninka.ru).

Aim and Scope

Discrete and Continuous Models and Applied Computational Science arose in 2019 as a continuation of RUDN Journal of Mathematics, Information Sciences and Physics. RUDN Journal of Mathematics, Information Sciences and Physics arose in 2006 as a merger and continuation of the series "Physics", "Mathematics", "Applied Mathematics and Computer Science", "Applied Mathematics and Computer Mathematics".

Discussed issues affecting modern problems of physics, mathematics, queuing theory, the Teletraffic theory, computer science, software and databases development.

It's an international journal regarding both the editorial board and contributing authors as well as research and topics of publications. Its authors are leading researchers possessing PhD and PhDr degrees, and PhD and MA students from Russia and abroad. Articles are indexed in the Russian and foreign databases. Each paper is reviewed by at least two reviewers, the composition of which includes PhDs, are well known in their circles. Author's part of the magazine includes both young scientists, graduate students and talented students, who publish their works, and famous giants of world science.

The Journal is published in accordance with the policies of COPE (Committee on Publication Ethics). The editors are open to thematic issue initiatives with guest editors. Further information regarding notes for contributors, subscription, and back volumes is available at http://journals.rudn.ru/miph.

E-mail: miphj@rudn.ru, dcm@sci.pfu.edu.ru.

EDITORIAL BOARD

Computer Design: *Anna V. Korolkova*, *Dmitry S. Kulyabov*

English text editors: *Nikolay E. Nikolaev*, *Ivan S. Zaryadov*, *Konstantin P. Lovetskiy*

Address of editorial board:
Ordzhonikidze St., 3, Moscow, Russia, 115419
Tel. +7 (495) 955-07-16, e-mail: publishing@rudn.ru
Editorial office:
Tel. +7 (495) 952-02-50, miphj@rudn.ru, dcm@sci.pfu.edu.ru
site: http://journals.rudn.ru/miph

Paper size 70×100/16. Offset paper. Offset printing. Typeface "Computer Modern".
Conventional printed sheet 6,93. Printing run 500 copies. Open price. The order 21.
PEOPLES' FRIENDSHIP UNIVERSITY OF RUSSIA
6 Miklukho-Maklaya St., 117198 Moscow, Russia
Printed at RUDN Publishing House:
3 Ordzhonikidze St., 115419 Moscow, Russia,
Ph. +7 (495) 952-04-41; e-mail: publishing@rudn.ru

Discrete & Continuous Models
& Applied Computational Science
ISSN 2658-7149 (online), 2658-4670 (print)
2023, 31 (1) 4
http://journals.rudn.ru/miph

Contents

Migran N. Gevorkyan, Anna V. Korolkova, Dmitry S. Kulyabov, Julia language features for processing statistical data 5

Irina I. Vasilyeva, Anastasia V. Demidova, Olga V. Druzhinina, Olga N. Masina, Construction, stochastization and computer study of dynamic population models "two competitors – two migration areas" 27

Kouame A. Brou, Ivan V. Smirnov, Causality relationship between foreign direct investments and economic improvement for developing economies: Russia case study 46

Nikolay A. Kolpakov, Alexey I. Molodchenkov, Anton V. Lukin, Methods of extracting biomedical information from patents and scientific publications (on the example of chemical compounds) 64

Nikolay Yu. Kravchenko, Sergey S. Kovtunov, Studying the mechanism of electric explosion of metal conductors 75

Discrete & Continuous Models
& Applied Computational Science
ISSN 2658-7149 (online), 2658-4670 (print)

2023, 31 (1) 5–26
http://journals.rudn.ru/miph

Research article

UDC 537.8:512.723
DOI: 10.22363/2658-4670-2023-31-1-5-26
EDN: VNJCSU

Julia language features for processing statistical data

Migran N. Gevorkyan[1],
Anna V. Korolkova[1], **Dmitry S. Kulyabov**[1,2]

[1] *Peoples' Friendship University of Russia (RUDN University),*
6, Miklukho-Maklaya St., Moscow, 117198, Russian Federation
[2] *Joint Institute for Nuclear Research,*
6, Joliot-Curie St., Dubna, Moscow Region, 141980, Russian Federation

(received: January 16, 2023; revised: March 13, 2023; accepted: April 10, 2023)

Abstract. The Julia programming language is a specialized language for scientific computing. It is relatively new, so most of the libraries for it are in the active development stage. In this article, the authors consider the possibilities of the language in the field of mathematical statistics. Special emphasis is placed on the technical component, in particular, the process of installing and configuring the software environment is described in detail. Since users of the Julia language are often not professional programmers, technical issues in setting up the software environment can cause difficulties that prevent them from quickly mastering the basic features of the language. The article also describes some features of Julia that distinguish it from other popular languages used for scientific computing. The third part of the article provides an overview of the two main libraries for mathematical statistics. The emphasis is again on the technical side in order to give the reader an idea of the general possibilities of the language in the field of mathematical statistics.

Key words and phrases: Julia programming language, statistic processing

1. Introduction

In this paper we give a brief overview of Julia [1] programming language capabilities in the field of mathematical statistics. Julia is a fast compiled language with dynamic typing, originally developed for scientific computing. The language is relatively new, however, it has already reached version 1.8 and the core of the language is quite stable. An impressive number of modules have been created for Julia and several books have been written [2–4].

There are a number of arguments in favor of learning and using the Julia language:

— Just-in-Time compilation (JIT) [5] allows you to simultaneously achieve high performance and ease of use of the interpreted language. Single-threaded programs in Julia have the performance of programs in C/C++ and Fortran [6] and significantly exceed the interpreted languages, such as R, Python, Matlab, SciLab, etc.

— The syntax of Julia is simple and for researchers familiar with Python, Fortran and R languages, it will not be difficult to master it at a basic level in the shortest possible time.
— The language has built-in extensive capabilities for parallel and distributed computing, which are constantly being refined.

The authors tend to give a general idea of Julia language's available capabilities in the field of mathematical statistics and demonstrate a number of examples that allow one to quickly grasp the features of the language and move on to use it. At the beginning of the article we give a step-by-step description of the configuration of the working environment for Unix-type systems (macOS, GNU/Linux) and Windows. We do not give a consistent description of the syntax of the language, but focus on some specific features (dynamic dispatching, custom data types) that distinguish Julia from most popular programming languages.

Libraries for mathematical statistics for Julia are combined under the general name Julia Statistics [7, 8] and a separate section is allocated for them on the official forum of language developers [9]. In the main part of this paper, we give an overview of the modules `StatsBase` and `Distributions` [10, 11], comparing their functionality with the libraries of the `R` language and the `scipy.stats` library of Python [12].

2. Installation and configuration of Julia environment

There are several ways of programs development in Julia languages.

— Using REPL-shell (read-eval-print loop) in interactive mode, by running the `julia` command from the terminal and entering instructions that will be executed immediately, and the user will see the returned result.
— By saving the program source code to files with the extension `jl` and then passing them for compilation and launching to the julia JIT compiler (same `julia` command).
— By using interactive shells, such as Jupiter Notebook [13] and Pluto [14].

We will describe the process of Julia installation, as well as Jupiter interactive shell and all necessary modules in the GNU/Linux and Windows environments. The installation does not require superuser rights and it can be performed remotely by connecting via ssh, which can be convenient if calculations are supposed to be performed on a remote server.

2.1. Installing Julia and the necessary packages

On the official Julia website, in the `downloads` section, binary files for many systems are presented. Download the archive for the 64-bit version of GNU/Linux:

```
wget https://julialang-s3.julialang.org/bin/linux/x64/1.8/juli␣
↪ a-1.8.5-linux-x86_64.tar.gz
↪ --no-check-certificate
```

Please note that the url may change, as it clearly indicates the current version of the Julia distribution. Extract the files from the downloaded archive:

```
tar -xvzf julia-1.8.5-linux-x86_64.tar.gz
```

The directory `julia-d386e40c17` will be created (or with another alphanumeric combination), which we will rename to just `julia`:

```
mv julia-d386e40c17/ julia
```

```
export PATH="~/julia/bin:$PATH"
```

In the case of the Windows operating system, we will describe the installation of the portable version. In the same section of the official website, download the 64-bit (portable) version for Windows and unpack the archive, for example, into the following directory:

```
E:\Program Files\julia
```

In this directory, we will create the folder `depot`, and in it, the folder `config`, in which we will create an empty text file `startup.jl`, which we will need next. The Julia directory `depot` will host an index of modules from the official repository, as well as installed modules and additional libraries. The location of this directory is non-standard and in order for the Julia JIT compiler to recognize it correctly at startup, you should create an environment variable `JULIA_DEPOT_PATH` and assign it a value:

```
E:\Program Files\julia\depot
```

We also added the path to the Julia JIT compiler to the variable `PATH` (executable file julia.exe)

```
E:\Program Files\julia\bin
```

After the installation is complete, run the Julia command shell. To do this, run the command `julia` in the console. In the case of Windows, it is recommended to use PowerShell or the new Windows Terminal application [15]. After launching, press the key `]` and switch to package management mode, where you run the command `update`, which will download the package index. You can also immediately install the necessary packages using the command `add`, for example

```
add StatsBase Distributions Pluto Plots
```

The built-in package manager saves all package-related files to the storage directory (depot), which is pointed to by the environment variable `JULIA_DEPOT_PATH`. No other directories are involved.

On Unix systems — if the variable `JULIA_DEPOT_PATH` is not defined — the corresponding directory is created in the user directory and is called `.julia`. In it, you should also manually create a directory `config` with the configuration file `startup.jl`.

At this stage, the installation and configuration of the compiler is complete and we will proceed to the installation of additional tools that may be needed to write programs on Julia.

2.2. Jupyter installation

Julia language code can be executed in the Jupyter environment, for which you should install the Jupyter Notebook kernel, which is included in the package `IJulia`. During the installation, the built-in Julia package manager automatically downloads the Python distribution miniconda [16], places it in the storage directory and installs with its help all the necessary python

packages, including Jupyter Notebook. Since most Julia users probably already have a Python distribution installed on the system, we will show you how to use the already installed Jupyter in Julia. Even if there is no Python distribution in the system, it seems more practical to install it separately, as this will allow better control of the packages used.

Next, we will describe the process of installing Jupiter using the Miniconda distribution into the user's local directory. Download the installation script from the official website:

```
wget https://repo.continuum.io/miniconda/Miniconda3-latest-Lin
↪ ux-x86_64.sh
```

and start the installation process:

```
bash Miniconda3-latest-Linux-x86_64.sh
```

During the installation process, you must read and accept the license agreement by using the key `Enter` to scroll through the text and typing the word `yes` to accept it. After that, the installer will prompt you to select the directory where the distribution directory tree will be copied. By default, this is the directory `miniconda3` in the user's home directory. Let's leave it unchanged, for which you should press `Enter`. The process of downloading the necessary files will begin, after which the installer will offer to add the path to the Python interpreter to the environment variable `PATH`. You should agree by typing the word `yes` and pressing `Enter`. Then check that the following line has been added to the file `.bashrc` located in the home directory:

```
export PATH="~/miniconda3/bin:$PATH"
```

where `~/miniconda3/bin` is the path to the miniconda directory. If the installer did not add this path automatically, you can do it manually.

After the anaconda installation is completed, the command `conda` will be available with which you can manage the installed Python modules. Let's use this command to install the modules we need (Numpy, SciPy, Matplotlib and Jupiter):

```
conda install numpy matplotlib scipy jupyter
```

The process of downloading and unpacking the required files can take considerable time, and after completion, the miniconda directory will occupy about 2.5 GB of disk space. To check the correctness of the installation, run Jupiter by running the following command:

```
jupyter notebook --notebook-dir=~ --port=7000
```

The interactive shell session will start. If launched on a local computer, a browser will automatically open with a list of files and directories of the home directory (option `--notebook-dir=~`). If you run it on a remote computer, you should add the option `--no-browser`, then you can connect to the session that has started remotely by entering the address of the remote computer into the local network in the browser address bar or organize an ssh tunnel.

Now, in the file `startup.jl`, which we previously created, but left empty, we should add local environment variables that will point to the locations of the executable files of the python interpreter and the jupyter shell:

```
ENV["PYTHON"] = "~/miniconda3/bin/python"
ENV["JUPYTER"] = "~/miniconda3/bin/jupyter"
```

Note that the variable `ENV` is a dictionary to which, when the compiler is started, system and user environment variables are added, as well as a number

of local Julia parameters. This dictionary is available in any julia program for reading and modification, which we used by adding two new keys to it.

After that, run REPL Julia with the command `julia` and install the necessary packages:

```
add PyCall PyPlot IJulia
```

During the installation process, the package manager will see that the variables `PYTHON` and `JUPITER` have been assigned values and the local copy of miniconda will not be installed.

After the installation is completed, when Jupiter Notebook is launched, the Julia kernel will be available and it will be possible to create and open interactive notebooks with scripts in the Julia language.

In addition to `Julia`, we also installed the package `PyCall`, which greatly simplifies calling functions from python modules, and the package `PyPlot`, which makes it possible to use the library `Matplotlib` in Julia. In this article, we will not use the capabilities of these packages, but for those users who are used to standard Python scientific libraries, they may be useful, since they transfer the usual functionality to Julia.

2.3. Pluto shell as an alternative to Jupiter

Using Jupyter with the Julia language has at first glance an unobvious drawback associated with a fundamental feature of the architecture of the language itself — multiple dispatching of functions. In the case of Jupyter, it manifests itself as follows: when initially creating a function in a separate cell and executing this cell once, no problems arise, however, if the programmer decides to change the body of this function without changing the signature of the arguments, then re-executing the cell with the modified code will lead to an error. If the list of arguments has been modified, there will be no errors during execution, but a new method will be created or, in other terminology, an overloaded version of the function will be created. This difficulty can be overcome by restarting the kernel, but this makes the process uncomfortable if the cells contain resource-intensive calculations, which will have to be done again every time. There will definitely be such cells, since the initial initialization of graphical libraries for data visualization in Julia is extremely slow, and the main advantage of Jupyter is precisely the interactive display of the results of various data visualization.

The Julia development community has created an alternative shell called `Pluto.jl`. At the moment, the repository of this package [17] ranks second in the number of stars on GitHub, second only to the Julia compiler itself [18].

Shell `Pluto.jl` is generally similar to `Jupyter`, but has two key differences:

— reactivity (reactive);
— no hidden states (no hidden workspace state).

Reactivity lies in the fact that all cells of the interactive notebook are immediately restarted if the variables on which they depend are modified, even if these variables are contained in other cells.

The absence of hidden states means that if a cell is deleted, then all the variables, functions and data structures contained in it are deleted from memory and become inaccessible. It is also impossible to redefine variables and functions in neighboring cells, which removes the problem with implicit function overloading.

To install `Pluto`, just run `add Pluto` in package management mode in Julia REPL. No additional manipulations are required, since `Pluto` is written in Julia only. At the time of writing, this package has reached version 0.19.22, but it works quite stably. Among the disadvantages, it can be noted that the interface is too minimalistic, as well as the demands on the amount of RAM.

To run the shell in Julia REPL mode, follow these instructions

```
import Pluto
Pluto.run()
```

At the same time, the browser will immediately be launched with a welcome interface, and a link will be displayed in the console, which can be used for remote connection.

Let's note some features of the interactive notepad. To store the contents of the notebook, a simple text file with the extension `jl` is used, the entire code of the cells is stored as a regular code in the Julia language, and standard code comments are used to store meta information. This makes it possible to execute Pluto notebooks like regular Julia programs, passing them to the JIT compiler for execution.

Pluto has built-in support for local package environments. It automatically detects the packages used by looking at instructions `used` and `import` and downloading the necessary packages to the local directory. This is useful if you need to transfer the created notebooks to third-party users, as well as fix specific versions of libraries. However, this behavior can be disabled if desired, for which the following code should be added to the first cell

```
begin
  using Pkg
  Pkg.activate()
end
```

This command will disable the local package manager and Pluto will use the standard Julia environment that was created when the package update was initially launched.

This code snippet illustrates another feature of Pluto which is called reactivity. By default, each cell can contain only one line of code. In the case of several lines, they must be framed with the construction `begin` ... `end`. This may cause some inconvenience, but the shell itself determines such cells and offers to automatically insert `begin` and `end`.

Pluto notepad allows you to add cells with comments in markdown format in combination with LaTeX formulas. Unlike jupiter notebooks in Pluto, these are not special cells, but standard ones with a multiline string preceded by the `md` modifier, for example:

```
md"""# Header
The text of the comment and the equation $\dot{x} = f(x)$
"""
```

Finally, we note that Pluto includes the module `PlutoUI`, which allows you to create interactive graphical interface elements such as sliders, drop-down lists, text input fields, etc. and bind variables to them. This allows you to add interactivity to the notebooks being created, which is useful, for example, for selecting parameters for certain functions. Due to reactivity, when changing the values of variables, all graphs that depend on them will also be rebuilt.

3. The main features of the Julia language

The Julia language was originally created for the field of scientific programming and its syntax is very similar to the syntax of the Fortran and Python languages, which are well known to specialists in scientific computing. However, at the same time, it contains some specific features, and without knowing them, it will be difficult to use third-party libraries effectively. We will illustrate all the features with examples from probability theory and mathematical statistics.

3.1. Custom data structures

One of the distinctive features of the Julia language is the high performance of data types created by the user himself. In many modules there is large number of custom data types and functions.

A composite data type in the first approximation resembles a structure from the C-language. It is specified using the construct `struct`, inside which the fields of the structure with the type annotation are listed. As an example, consider setting a structure that stores the parameters of a normal distribution.

```
"Normal distributions"
struct Normal
    "first moment"
    μ::Real
    "standard deviation"
    σ::Real
end
```

Let's list some important features.

— Since Julia provides full Unicode encoding support, Greek letters and other symbols that are standard for mathematical formulas can be used as field designations.

— A composite type and its fields can be provided with documentation lines that explain the purpose of the structure and its fields. These strings are similar to Python doc-strings, with the difference that they can be supplied to almost any object and they must be specified before, not after the declaration.

— Julia is a dynamically typed language, but it supports type annotations that can be used by the compiler for code optimization and to limit the types of variables passed to functions when they are called and to structures when they are initialized.

After defining the structure, you can create objects of type `Normal` using the default constructor.

```
N = Normal(0, 1)
@show typeof(N)
@show N.μ, N.σ
```

The macro `@show` prints the line of code that is passed to it and the result of executing this line of code. So, in the example above, the following will be printed to standard output:

```
typeof(N) = Normal
(N.μ, N.σ) = (0, 1)
```

The default constructor is created automatically, but it can be set explicitly in the body of the structure, for example, if you need to limit the scope of acceptable values of the fields of the structure. After the checks, it is necessary to allocate memory for the fields of the structure using a special function `new`.

```
struct Normal
  μ::Real
  σ::Real
  function Normal(μ, σ)
    if σ == 0
      throw(ArgumentError("σ != 0"))
    end
    return new(μ, σ)
  end
end
```

Only one main constructor can be defined in the structure body. If you need to define additional constructors, they should be set outside the structure. So, you can define a constructor without arguments, which will set the parameters of the standard normal distribution.

```
function Normal()
  return Normal(0.0, 1.0)
end
```

3.2. Multiple dispatch

Julia implements a multiple dispatching mechanism [5, 19, 20], which, according to the developers, is a more flexible mechanism compared to the object-oriented approach applied to mathematical applications.

Each function in Julia can have many implementations called *methods*. Implementations have the same name, but differ from each other both in the number of arguments and their types. When calling a function, the compiler analyzes the arguments passed to it and calls the desired implementation. Various operators such as `+`, `-` are also functions and can be overridden for any new data type.

To illustrate multiple dispatching, we additionally define a structure that stores the parameters of the exponential distribution:

```
struct Exponential <: Distribution
  λ::Real
  function Exponential(λ)
    if λ <= 0
      throw(ArgumentError("λ > 0"))
    end
    return new(λ)
  end
end
```

Now we implement two functions that calculate the PDF of normal and exponential distributions:

```
function pdf(d::Normal, x)
    return 1/(sqrt(2*π)*d.σ) * exp(-(x-d.μ)^2 / (2*d.σ^2))
end
```

```
function pdf(d::Exponential, x)
    return d.λ * exp(-d.λ*x)
end
```

It should be noted that the first arguments of the function are provided with type annotations. This is done so that the compiler can call the desired implementation depending on the type of the first argument:

```
N = Normal(0, 1)
E = Exponential(2)
@show pdf(N, 3) # <- call of the implementation for the normal
 ↪ distribution
@show pdf(E, 2) # <- call of the implementation for the
 ↪ exponential distribution
```

In addition, Julia allows you to automatically vectorize a scalar function, that is, apply it to each element of some array, without having to implement an additional method. To do this, it is enough to use a special syntax:

```
pdf.(N, [1, 2, 3, 4, 5])
```

To achieve a similar effect, R uses the function `Vectorize`, and Python uses `map` or a list assembly.

4. Module StatsBase.jl overview

In the module `StatsBase.jl` [10] implement basic functions for working with statistical samples presented as one-dimensional arrays. Due to the ease of use of most functions, we will not dwell on examples, but will give only short description of the main functionality of this module.

- Vectors of the sample weight coefficients (weight vectors).
- Functions that calculate the mean (geometric, harmonic, power and weighted arithmetic mean).
- The simplest statistical functions.
  - Moments that take into account the vectors of weight coefficients: mathematical expectation, variance, standard deviation, skewness coefficient, kurtosis coefficient and central moments of arbitrary order.
  - Standardized score (Z-score).
  - Entropy calculations, such as standard, Rényi (generalized) entropy, crossentropy, Kullback-Leibler divergence distance.
  - Quantiles and mods.
- Robust statistics: truncation and winsorization of the sample.
- Comparing two samples, by calculating different discrete metrics.
- Calculation of scattering, covariance, and correlation matrices.
- Functions that calculate the frequency of occurrence of a particular value in the sample.
- Calculation of histograms.
- Autocorrelation and autocovariance.

Functions from the module `StatsBase.jl` is actively used in other modules, so it is included in the list of dependencies of most statistical libraries created for `Julia`.

5. Module Distributions.jl overview

5.1. Brief overview of the module

The module `Distributions.jl` [21] implements functions and methods related to probability distributions (mainly one-dimensional discrete and continuous, as well as a small number of multidimensional ones).

— Probability distribution Functions (CDF) and probability distribution density functions (PDF).
— Functions for calculating statistical characteristics of distributions (expectation, variance, moments, modes, quantiles, kurtosis, etc.).
— Characteristic functions of distributions and generating functions of moments.
— Methods for selecting distribution parameters based on statistical data (distribution fitting) by the maximum likelihood method (Maximum Likelihood) and the Sufficient Statistics method (Sufficient Statistics).

5.2. Module installation

In order to use the module `Distributions.from`, it must first be installed using the command `Pkg.add("Distribution.from")`. After that, it can be imported using the instructions `using` or `import`. We use the second method to avoid mixing the module namespaces `Distributions.jl` with the global scope.

```
import Distributions
const dist = Distributions
```

Now `dist` will serve as a short synonym for Distribution and all functions and variables defined in the module name area will be accessible via the period `operator.`.

5.3. Creating a probability distribution

Since Julia's custom data types are not inferior in performance to the built-in data types, in the module `Distributions.jl`, probability distributions are implemented as additional data types. For example, to set a normal distribution, you should call the constructor `Normal`, passing to it two parameters: μ and σ are the mathematical expectation and the standard deviation:

```
μ, σ = 0.0, 1.0 # location, scale
Normal = dist.Normal(μ, σ)
```

The variable `Normal` will now have the type `Distributions.Normal{Float64}`. If you call the constructor without arguments, then the standard normal distribution will be set, with $\mu = 0$ and $\sigma = 1$.

Similarly, other distributions can be set, for example, the Beta distribution:

```
# Beta distribution
α, β = 1.0, 1.0 # shape
Beta = dist.Beta(α, β)
```

Complete lists of discrete and continuous distributions are given in the tables 1 and 2. On the official documentation page [11] there is a description of almost all implemented distributions, from which you can find out what

parameters and in what order you need to pass to the constructor and what default values are provided for these parameters. Basically, developers adhere to the established notation in the literature.

Table 1
List of one-dimensional discrete distributions implemented in the module `Distribution.jl`

№	Function	Description
1	`Bernoulli`	Bernoulli distribution
2	`BetaBinomial`	Beta-Binomial distribution
3	`Binomial`	Binomial distribution
4	`Categorical`	Categorical distribution
5	`DiscreteUniform`	Discrete Uniform distribution
6	`Geometric`	Geometric distribution
7	`Hypergeometric`	Hypergeometric distribution
8	`NegativeBinomial`	Negative Binomial distribution
9	`Poisson`	Poisson distribution
10	`PoissonBinomial`	Poisson-Binomial distribution
11	`Skellam`	Skellam distribution

Table 2
A complete list of distributions implemented in the `Distribution.jl` module and similar functions from the `scipy.stats` library from various modules of the `R` language

No	Distribution	Function name		
		`Distributions.jl`	`scipy.stats`	`R`
1	Arcsine	`Arcsine`	`arcsin`	`[distr]`
2	Beta	`Beta`	`beta`	`beta`
3	Beta Prime	`BetaPrime`	`betaprime`	`{VGAM}`
4	Biweight	`Biweight`	-	-
5	Cauchy	`Cauchy`	`cauchy`	`cauchy`
6	Chi	`Chi`	`chi`	`[Runuram]`
7	Chisq	`Chisq`	`chi2`	`chisq`
8	Cosine	`Cosine`	`cosin`	-
9	Erlang	`Erlang`	`erlang`	`gamma`

Table 2
A complete list of distributions implemented in the `Distribution.jl` module and similar functions from the `scipy.stats` library from various modules of the `R` language (continuation)

No	Distribution	Function name		
		`Distributions.jl`	`scipy.stats`	`R`
10	Epanechnikov	`Epanechnikov`	`-`	`[epandist]`
11	Exponential	`Exponential`	`expon`	`exp`
12	Fisher Distribution	`FDist`	`f`	`f`
13	Frechet	`Frechet`	`frechet_r`, `frechet_l`	`[VGAM]`
14	Gamma	`Gamma`	`gamma`	`gamma`
15	Generalized Extreme Value	`GeneralizedExtreme Value`	`genextreme`	`[spatial Extremes]`
16	Generalized Pareto	`GeneralizedPareto`	`genpareto`	`[evd]`
17	Gumbel	`Gumbel`	`gumbel_r`	`[VGAM]`
18	Inverse Gamma	`InverseGamma`	`invgamma`	`[extra Distr]`
19	Inverse Gaussian	`InverseGaussian`	`invgauss`	`[statmod]`
20	Kolmogorov	`Kolmogorov`	`-`	`[kolmim]`
21	K-S test	`KSDist`	`kswobign`	`[kolmin]`
22	K-S test one side	`KSOneSided`	`ksone`	`[kolmim]`
23	Laplace	`Laplace`	`laplace`	`[distr]`
24	Levy	`Levy`	`levy`	`[VGAM]`
25	Logistic	`Logistic`	`logistic`	`[VGAM]`
26	Log-Normal	`LogNormal`	`lognormal`	`lnorm`
27	Noncentral Beta	`NoncentralBeta`	`-`	`beta`
28	Noncentral Chisq	`NoncentralChisq`	`ncx2`	`chisq`
29	Noncentral Fisher	`NoncentralF`	`ncf`	`sadists`
30	Noncentral Student	`NoncentralT`	`nct`	`sadists`
31	Normal	`Normal`	`norm`	`norm`

Table 2
A complete list of distributions implemented in the `Distribution.jl` module and similar functions from the `scipy.stats` library from various modules of the `R` language (continuation)

No	Distribution	Function name		
		`Distributions.jl`	`scipy.stats`	`R`
32	NormalCanon	`NormalCanon`	-	`norm`
33	Normal Inverse Gaussian	`NormalInverseGaussian`	-	`[ghyp]`
34	Pareto	`Pareto`	`pareto`	`[VGAM]`
35	Rayleigh	`Rayleigh`	`rayleigh`	`[VGAN]`
36	Wigner semicircle distribution	`Semicircle`	`semicircular`	
37	SymTriangularDist	`SymTriangularDist`	-	`[triangle]`
38	Student	`TDist`	`t`	`t`
39	TriangularDist	`TriangularDist`	`triang`	`[triangle]`
40	Triweight	`Triweight`	-	-
41	Uniform	`Uniform`	`uniform`	`unif`
42	Von Mises	`VonMises`	`vonmises`	`[mov MF]`
43	Weibull	`Weibull`	`weibull_min`, `weibull_max`	`weibull`

5.4. Calculation distributions characteristics

A number of functions listed in the table 3 are intended to obtain characteristics of theoretical distributions. As an argument, they take a variable of type `Distribution` and, depending on the distribution, return the requested parameters. If this type of distributions does not have one or another parameter, an exception is thrown. For example, the `dof` function returns the number of degrees of freedom of the distribution, so that for a normal distribution it will end with an exception thrown, and for a Student distribution it will return the value of the requested parameter.

Another set of functions is used to calculate statistical characteristics of both theoretical distributions and empirical data. A complete list of these functions is given in the table 4. Some of them are defined in the scope of the Julia language base module (`Base`), as they overload standard functions such as `mean`, `median`, etc. More specific functions are defined in the scope of the module `Distributions`.

A number of functions can be used to calculate the statistical characteristics of an empirical sample (i.e., take an array of random numbers as the first

argument). For such functions in the table 4 in the column **v** there is a mark `+`. Note that the function `quantile` supports an array as the first argument, while the function `cquantile` does not.

Table 3
List of functions for getting distribution parameters from the module `Distribution.jl`

Function	**Description**
`params(d)`	return description's parameters
`succprob(d)`	not implemented yet
`failprob(d)`	not implemented yet
`scale(d)`	return `scale` parameter (if not, throw error)
`location(d)`	return `location` parameter (if not, throw error)
`shape(d)`	return `shape` parameter (if not, throw error)
`rate(d)`	return `rate` parameter (if not, throw error)
`ncategories(d)`	return number of categories
`ntrials(d)`	get the number of trials
`dof(d)`	return degree of freedom

Consider an example. So the function `mean` can be used to find the mathematical expectation if one passes as an argument an object representing some distribution. Or to calculate the average, if one passes an array of numbers (representing a statistical sample).

```
mean(Normal) # return 0
mean([1, 2, 3, 4, 5]) # return 3
```

5.5. One dimensional distributions

In the Distributions package.jl implemented 11 discrete distributions (table 1) and 43 continuous one-dimensional distributions (see table 2). This table gives a summary of the functions from Distributions.jl and its equivalent functions from the library `scipy.stats` and language packages `R`. This will allow readers familiar with `R` or `scipy.stats` to orient themselves.

5.6. Implementations of functions for calculating statistical characteristics for different distributions

The table 5 shows the results of testing functions that calculate statistical characteristics for various distributions. In the table, the sign `+` means that the function is implemented for this distribution, and the sign `-` means that there is no implementation. Note that these are implementations specifically for theoretical distributions. It should also be borne in mind that the lack of implementation may mean that there is no exact analytical formula for this distribution.

Table 4
List of functions for calculating statistical characteristics of distributions from the module `Distribution.jl`

Function	**Description**	**V**
`maximum(d)`	maximum value	+
`mininum(d)`	minimum value	+
`mean(d)`	mathematical expectation	+
`var(d)`	variation	+
`std(d)`	standard deviation	+
`median(d)`	median	+
`dist.mode(d)`	mode	+
`dist.skewness(d)`	asymmetry coefficient	+
`dist.kurtosis(d)`	kurtosis coefficient	+
`dist.isplatykurtic(d)`	checks the kurtosis coefficient ($> 0, < 0, = 0$)	-
`dist.isleptokurtic(d)`		-
`dist.ismesokurtic(d)`		-
`dist.entropy(d)`	entropy	+
`dist.pdf(d, t)`	PDF	-
`dist.cdf(d, t)`	CDF	-
`dist.logpdf(d, t)`	ln from PDF	-
`dist.logcdf(d, t)`	ln from CDF	-
`dist.mgf(d, t)`	generating function of moments	-
`dist.cf(d, t)`	characteristic distribution function	-
`quantile(d, q)`	quantile q	+
`cquantile(d, q)`	complementary quantile $1 - q$	-
`invlogcdf(d, t)`	inverse function for logcdf	-
`invlogccdf(d, t)`	inverse function for logccdf	-

Table 5

Implementation of functions for calculating statistical characteristics of distributions from the module `Distribution.jl`.
(Notation: `+` — method is implemented, `-` — method is not implemented (MethodError), `@` — domain error, `o` — any other error)

Distribution	maximum	minimum	mean	var	std	median	mode	skewness	kurtosis	isplatykurtic	isleptokurtic	ismesokurtic	entropy	pdf	cdf	logpdf	logcdf	mgf	cf	quantile	cquantile	invlogcdf	invlogccdf
Arcsine	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	+
Beta	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	+
Beta Prime	+	+	+	+	+	+	+	+	-	-	-	-	-	+	+	+	+	-	-	+	+	+	+
Biweight	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+	+	+	+	+	+	+	+	+
Cauchy	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Chi	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	+
Chisq	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Cosine	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+	+	+	-	-	+	+	o	o
Erlang	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Epanechnikov	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+	+	+	+	+	+	+	+	+
Exponential	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	@	+
Fisher Distribution	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	+
Frechet	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	@
Gamma	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Generalized Extreme Value	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	@

Table 5
Implementation of functions for calculating statistical characteristics of distributions from the module `Distribution.jl` (continuation)

Distribution	maximum	minimum	mean	var	std	median	mode	skewness	kurtosis	isplatykurtic	isleptokurtic	ismesokurtic	entropy	pdf	cdf	logpdf	logcdf	mgf	cf	quantile	cquantile	invlogcdf	invlogccdf
Generalized Pareto	+	+	+	+	+	+	-	+	+	+	+	+	-	+	+	+	+	-	-	+	+	+	-
Gumbel	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	@	@
Inverse Gamma	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	@	+	+	+	+	+
Inverse Gaussian	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+	+	+	-	-	+	+	+	+
Kolmogorov	+	+	+	+	+	+	+	-	-	-	-	-	-	+	+	+	+	-	-	+	+	+	+
K-S	+	+	-	-	-	-	-	-	-	-	-	-	-	-	+	+	+	-	-	-	-	-	-
K-S one side	+	+	-	-	-	-	-	-	-	-	-	-	-	-	+	-	+	-	-	-	-	-	-
Laplace	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	@	@
Levy	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	@
Logistic	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	@	@
Log-Normal	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	+
Noncentral Beta	+	+	-	-	-	+	-	-	-	-	-	-	-	+	+	+	+	-	-	+	+	+	+
Noncentral Chisq	+	+	+	+	+	+	-	+	+	+	+	+	-	+	+	+	+	+	+	+	+	+	+
Noncentral Fisher	+	+	+	+	+	+	-	-	-	-	-	-	-	+	+	+	+	-	-	+	+	+	+
Noncentral Student	+	+	+	+	+	+	-	-	-	-	-	-	-	+	+	+	+	-	-	+	+	+	+

Table 5
Implementation of functions for calculating statistical characteristics of distributions from the module `Distribution.jl` (continuation)

Distribution	maximum	minimum	mean	var	std	median	mode	skewness	kurtosis	isplatykurtic	isleptokurtic	ismesokurtic	entropy	pdf	cdf	logpdf	logcdf	mgf	cf	quantile	cquantile	invlogcdf	invlogccdf
Normal	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
NormalCanon	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	+
Normal Inverse Gaussian	+	+	+	+	+	-	+	+	+	+	+	+	-	+	-	+	-	-	-	-	-	-	-
Pareto	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	+	+
Rayleigh	@	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	@	@
Wigner semicircle distribution	+	+	+	+	+	+	+	+	-	-	-	-	+	+	+	+	+	-	-	+	+	+	+
SymTriangularDist	@	@	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	@	@
Student	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+	+	+	+
TriangularDist	@	@	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	@	@
Triweight	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+	+	+	+	+	+	+	+	+
Uniform	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Von Mises	-	-	+	+	+	+	+	-	-	-	-	-	+	+	+	+	+	-	+	-	-	-	-
Weibull	@	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	+	+	@	+

For those distributions for which at least one of the functions `PDF` and `CDF` is implemented, graphs are constructed (see, for example in figures 1 and 2).

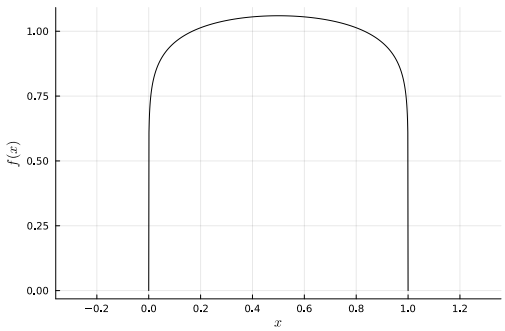

Figure 1. Example of Beta distribution PDF

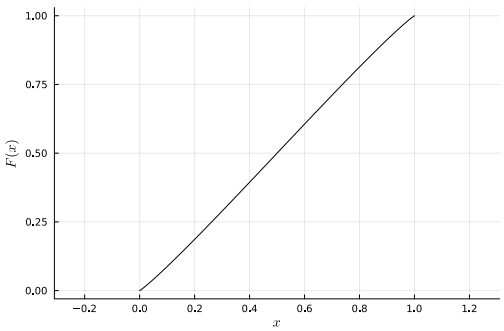

Figure 2. Example of Beta distribution CDF

5.7. The distribution parameters determination by the sample

To determine the parameters of the theoretical distribution over the sample, you can use the universal function `fit(d, x)`.

This function uses either the maximum likelihood method or the method of sufficient statistics in its work. The first method is implemented in the function `fit_me`, and the second one in the function `suffstats`. The developers recommend using the function `fit`, which chooses the optimal method itself. Consider an example of usage.

```
X = rand(Normal, 100)
# A normal distribution whose coefficients
# calculated based on a statistical sample
E_Normal = dist.fit(dist.Normal, X)
```

Selection of coefficients based on the sample is implemented only for a small number of distributions available in the module. Among which:

— Bernoulli distributions, discrete uniform, geometric, binomial, categorical and Poisson distributions;
— Beta, exponential, normal, gamma, Laplace, Pareto, uniform distributions.

As you can see, among the distributions there are no quite commonly used ones, such as the Weibull distribution or the lognormal distribution.

6. Conclusion

As a result of the review of the capabilities of the Julia language in the field of mathematical statistics, it can be concluded that in terms of the richness of functionality, it is still inferior to the specialized R language and the capabilities of Python libraries. However, it surpasses these languages in the speed factor, and the intensity of the Julia language development makes it possible to assume that the missing functionality will be implemented over time.

Acknowledgments

This paper has been supported by the RUDN University Strategic Academic Leadership Program.

References

[1] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM Review*, vol. 59, no. 1, pp. 65–98, Jan. 2017. DOI: 10.1137/141000671.

[2] B. Lauwens and A. Downey, *Think Julia*. O'Reilly Media, Inc., 2019.

[3] T. Kwong, *Hands-on design patterns and best practices with Julia*. Packt Publishing, 2020.

[4] C. T. Kelley, *Solving nonlinear equations with iterative methods, Solvers and Examples in Julia*. SIAM, 2022.

[5] J. Bezanson, J. Chen, B. Chung, S. Karpinski, V. B. Shah, J. Vitek, and L. Zoubritzky, "Julia: dynamism and performance reconciled by design," *Proceedings of the ACM on Programming Languages*, vol. 2, no. OOPSLA, pp. 1–23, Oct. 2018. DOI: 10.1145/3276490.

[6] M. N. Gevorkyan, A. V. Korolkova, D. S. Kulyabov, and K. P. Lovetskiy, "Statistically significant comparative performance testing of Julia and Fortran languages in case of Runge–Kutta methods," in *Numerical methods and applications. NMA 2018*, ser. Lecture Notes in Computer Science, G. Nikolov, N. Kolkovska, and K. Georgiev, Eds., vol. 11189, Cham: Springer International Publishing, 2019, ch. 45, pp. 400–407. DOI: 10.1007/978-3-030-10692-8_45.

[7] "JuliaStats, Statistics and machine learning made easy in julia." (2023), [Online]. Available: https://juliastats.org/.

[8] Y. Nazarathy and H. Klok, *Statistics with Julia, Fundamentals for Data Science, Machine Learning and Artificial Intelligence*. Springer International Publishing, 2021. DOI: 10.1007/978-3-030-70901-3.

[9] "Julia forums." (2023), [Online]. Available: https://discourse.julialang.org.

[10] "StatsBase.jl." (2023), [Online]. Available: https://github.com/JuliaStats/StatsBase.jl.

[11] "Distributions.jl." (2023), [Online]. Available: https://github.com/JuliaStats/Distributions.jl.

[12] C. Führer, J. E. Solem, and O. Verdier, *Scientific computing with Python, High-performance scientific computing with NumPy, SciPy, and pandas*, 2nd. Packt Publishing Ltd., 2021.

[13] D. Toomey, *Learning Jupyter*. Packt Publishing Ltd., 2016.

[14] "Pluto.jl — interactive Julia programming environment." (2023), [Online]. Available: https://plutojl.org/.

[15] "Windows terminal, console and command-line repo." (2023), [Online]. Available: https://github.com/microsoft/terminal.

[16] "Miniconda." (2023), [Online]. Available: https://docs.conda.io/en/latest/miniconda.html.

[17] "Pluto.jl GitHub." (2023), [Online]. Available: https://github.com/fonsp/Pluto.jl.

[18] "Julia GitHub." (2023), [Online]. Available: https://github.com/JuliaLang/julia.

[19] A. V. Korolkova, M. N. Gevorkyan, and D. S. Kulyabov, "Implementation of hyperbolic complex numbers in Julia language," vol. 30, no. 4, pp. 318–329, Dec. 2022. DOI: 10.22363/2658-4670-2022-30-4-318-329.

[20] R. Muschevici, A. Potanin, E. Tempero, and J. Noble, "Multiple dispatch in practice," in *OOPSLA'08: Proceedings of the 23rd ACM SIGPLAN conference on Object-oriented programming systems languages and applications*, ACM Press, Oct. 2008, pp. 563–582. DOI: 10.1145/1449764.1449808.

[21] M. Besançon, T. Papamarkou, D. Anthoff, A. Arslan, S. Byrne, D. Lin, and J. Pearson, "Distributions.jl: Definition and modeling of probability distributions in the JuliaStats ecosystem," *Journal of Statistical Software*, vol. 98, no. 16, pp. 1–30, 2021. DOI: 10.18637/jss.v098.i16.

For citation:

M. N. Gevorkyan, A. V. Korolkova, D. S. Kulyabov, Julia language features for processing statistical data, Discrete and Continuous Models and Applied Computational Science 31 (1) (2023) 5–26. DOI: 10.22363/2658-4670-2023-31-1-5-26.

Information about the authors:

Korolkova, Anna V. — Docent, Candidate of Sciences in Physics and Mathematics, Associate Professor of Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University) (e-mail: korolkova-av@rudn.ru, phone: +7(495) 952-02-50, ORCID: https://orcid.org/0000-0001-7141-7610)

Gevorkyan, Migran N. — Docent, Candidate of Sciences in Physics and Mathematics, Associate Professor of Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University) (e-mail: gevorkyan-mn@rudn.ru, phone: +7 (495) 955-09-27, ORCID: https://orcid.org/0000-0002-4834-4895)

Kulyabov, Dmitry S. — Professor, Doctor of Sciences in Physics and Mathematics, Professor at the Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University) (e-mail: kulyabov-ds@rudn.ru, phone: +7 (495) 952-02-50, ORCID: https://orcid.org/0000-0002-0877-7063)

УДК 537.8:512.723
DOI: 10.22363/2658-4670-2023-31-1-5-26
EDN: VNJCSU

Возможности языка Julia для обработки статистических данных

М. Н. Геворкян[1], **А. В. Королькова**[1], **Д. С. Кулябов**[1, 2]

[1] *Российский университет дружбы народов, ул. Миклухо-Маклая, д. 6, Москва, 117198, Россия*
[2] *Объединённый институт ядерных исследований, ул. Жолио-Кюри, д. 6, Дубна, Московская область, 141980, Россия*

Аннотация. Язык программирования Julia является специализированным языком для научных вычислений. Язык сравнительно новый, поэтому большинство библиотек для него находится в активной стадии разработки. В статье авторы рассматривают возможности применения языка в области математической статистики. Особый акцент делается на технической составляющей, в частности подробно описывается процесс установки и настройки программного окружения. Так как пользователи языка Julia зачастую не являются профессиональными программистами, технические моменты в настройке программного окружения могут вызывать у них трудности, препятствующие быстрому освоению базовых возможностей языка. В статье описываются некоторые особенности Julia, которые отличают его от других популярных языков, используемых для научных вычислений. Также даётся обзор двух основных библиотек для математической статистики. Упор опять-таки делается на технической стороне, чтобы дать читателю представление об общих возможностях языка в области математической статистики.

Ключевые слова: язык программирования Julia, обработка статистических данных

Discrete & Continuous Models
& Applied Computational Science
ISSN 2658-7149 (online), 2658-4670 (print)

2023, 31 (1) 27–45
http://journals.rudn.ru/miph

Research article

UDC 519.6:004.94
PACS 07.05.Tp, 02.60.Pn, 02.70.Bf
DOI: 10.22363/2658-4670-2023-31-1-27-45
EDN: VFNJWV

Construction, stochastization and computer study of dynamic population models "two competitors – two migration areas"

Irina I. Vasilyeva[1], **Anastasia V. Demidova**[2], **Olga V. Druzhinina**[3], **Olga N. Masina**[1]

[1] *Bunin Yelets State University, 28, Kommunarov St., Yelets, 399770, Russian Federation*
[2] *Peoples' Friendship University of Russia (RUDN University), 6, Miklukho-Maklaya St., Moscow, 117198, Russian Federation*
[3] *Federal Research Center "Computer Science and Control" of RAS, 44-2, Vavilov St., Moscow, 119333, Russian Federation*

(received: March 13, 2023; revised: March 28, 2023; accepted: April 10, 2023)

Abstract. When studying deterministic and stochastic population models, the actual problems are the formalization of processes, taking into account new effects caused by the interaction of species, and the development of computer research methods. Computer research methods make it possible to analyze the trajectories of multidimensional population systems. We consider the "two competitors – two migration areas" model, which takes into account intraspecific and interspecific competition in two populations, as well as bidirectional migration of both populations. For this model, we take into account the variability of the reproduction rates of species. A formalized description of the four-dimensional model "two competitors – two migration areas" and its modifications is proposed. Using the implementation of the evolutionary algorithm, a set of parameters is obtained that ensure the coexistence of populations under conditions of competition between two species in the main area, taking into account the migration of these species. Taking into account the obtained set of parameters, a positive stationary state is found. Two-dimensional and three-dimensional projections of phase portraits are constructed. Stochastization of the model "two competitors – two migration areas" is carried out based on the method of self-consistent one-step models constructing. The Fokker–Planck equations are used to describe the structure of the model. A transition to a four-dimensional stochastic differential equation in the Langevin form is performed. To carry out numerical experiments, a specialized software package is used to construct and study stochastic models, and a computer program based on differential evolution is developed. Algorithms for generating trajectories of the Wiener process and multipoint distributions and modifications of the Runge–Kutta method are used. In the deterministic and stochastic cases, the dynamics of the trajectories of population-migration systems is studied. A comparative analysis of deterministic and stochastic

models is carried out. The results can be used in modeling of different classes of dynamic systems.

Key words and phrases: population dynamics models, stochastic differential equations, one-step processes, stochastization, competition, migration, trajectory dynamics, projections of phase portraits, computer modeling, software package

1. Introduction

The classical Lotka–Volterra models [1, 2] are further developed in numerous papers by researchers [3–8]. Significant progress is associated with the analysis of dynamic models of ecological systems using the methods of the theory of stability of solutions of differential equations and optimization theory [3–6, 9–11]. It should be noted that when studying population models, the transition from the deterministic to the stochastic case is of great theoretical and applied interest [12–15].

Population dynamic models are characterized by the fact that when describing them, it is necessary to take into account various types of interaction in the population community, for example, intraspecific competition, interspecific competition, trophic interactions, migration, mutualism [16–18]. For example, research is being conducted related to the study of the properties of multidimensional ecological and demographic systems, taking into account competition and migration flows [19–22]. As the results show, the impact of migration can be significant, and the presence of migration flows leads to the emergence of new qualitative effects. The presence of migration flows in a population system is associated with an adaptive change in the behavior of an organism under changing environmental conditions, in particular, with a deterioration in the epidemiological situation or with an increase in population densities [23]. When constructing models, migration mechanisms are described using linear and nonlinear functions [15, 24–26]. The stability and qualitative behavior of population-migration models are considered in [18, 24, 26–29] and other papers. Despite a number of interesting results in the direction of studying systems with migration flows, there is a need to construct and research new models with migration.

As is known, one-dimensional Fokker–Planck equations are used in the construction of Gaussian stochastic models of small dimension. For multidimensional models, the simplest linear models with additive noise are most often used, however, this approach does not fully take into account stochastic processes in the system. A promising direction is stochastic modeling of dynamic systems based on the method of constructing self-consistent one-step models [30–32]. Using this method, we can perform an algorithmic transition to a stochastic model and evaluate the influence of stochastics on the qualitative properties of the model. This assessment is performed through a comparative analysis of deterministic and stochastic models with selected sets of parameters. When studying high-dimensional models, the choice of parameters can be carried out by applying evolutionary algorithms [12, 33–35]. Various systems of population dynamics (with competition, mutualism, migration) based on self-consistent models are considered in [12, 14, 28] and other papers.

Researchers consider various generalizations and modifications of the classical Lotka–Volterra models in the direction of increasing the dimension and constructing non-deterministic models. When considering such models, there is a need for computer research, taking into account the capabilities of high-level languages and applied mathematical packages [36–39]. Numerical analysis of behavior and computer studies of the dynamics of trajectories are associated, among other things, with new problems in the study of nonlinear processes, taking into account the processing of large data arrays under uncertainty. A software package is developed for stochastic modeling of various dynamic systems based on the method of constructing self-consistent one-step models [30, 31]. For the controlled case, a set of programs is proposed that combines randomization, optimization and machine learning [12].

Modeling of population-migration systems is carried out using various software that have a fairly effective set of tools for constructing computer models and conducting computational experiments [38, 40]. The use of applied mathematical packages and high-level programming languages makes it possible to study multidimensional population systems taking into account different types of intraspecific and interspecific interactions, as well as taking into account the variation of parameters and variables.

The three-dimensional model "predator-prey-one migration area" is considered in [29]. Four-dimensional population models with competition and one area of migration are studied in [20, 41]. This article is devoted to the study of such a four-dimensional population model of the type "two competitors – two migration areas", which takes into account changes in the reproduction rates of populations.

Section 2 of the paper considers the construction of the "two competitors – two migration areas" model with bidirectional migration (to two refuges) and its modifications. In particular, we offer a description of the model, in which the reproduction rate of population growth are different without varying the parameters of competition and migration. In Section 3, search for model parameters using an evolutionary algorithm is carried out. A study of a deterministic four-dimensional model is carried out, two-dimensional and three-dimensional projections of phase portraits are constructed. In Section 4, stochastic models "two competitors – two migration areas" are constructed using the method of constructing self-consistent stochastic models. In Section 5 the dynamics of trajectories for deterministic and stochastic models are studied. The results of computer experiments are presented and the interpretation of these results is given taking into account the comparison of stochastic and deterministic models. A software package developed in Python using the NumPy, SymPy, SciPy libraries is used as a tool for studying models.

2. Description of the deterministic model "two competitors – two migration areas" and its modifications

One of the basic population-migration models, taking into account competition and migration flows, is a three-dimensional model that describes the dynamics of two interrelated species. According to this model, the first species competes with the second species in the first area, taking into account the migration of the first species to the second area [19]. Four-dimensional

generalizations of this population-migration model are studied in [19, 21, 22, 42] and in other papers.

Next, we describe a four-dimensional model that takes into account the influence of interspecies competition in two populations with bidirectional migration of both populations. This model is given by a system of nonlinear differential equations of the form

$$\begin{aligned} \dot{x}_1 &= a_1 x_1 - p_{11} x_1^2 - p_{13} x_1 x_3 + \beta x_2 - \gamma x_1, \\ \dot{x}_2 &= a_2 x_2 - p_{22} x_2^2 + \gamma x_1 - \beta x_2, \\ \dot{x}_3 &= a_3 x_3 - p_{33} x_3^2 - p_{31} x_1 x_3 + \varepsilon x_4 - \delta x_3, \\ \dot{x}_4 &= a_4 x_4 - p_{44} x_4^2 + \delta x_3 - \varepsilon x_4. \end{aligned} \tag{1}$$

where x_1 is density of a competing population of the first species in the first area; x_2 is a population density of the first species in the second area (in the first refuge); x_3 is density of the competing population of the second species in the first area; x_4 is a population density of the second species in the third area (in the second refuge); a_i, $i = 1, 2, 3, 4$, are natural growth coefficients; p_{13}, p_{31} are coefficients of interspecific competition; p_{11}, p_{22}, p_{33}, p_{44} are coefficients of intraspecific competition; β, γ are coefficients of migration of a species between the first and second areas, with the second area is a refuge; δ, ε are coefficients of species migration between the first and third areas, with the third area is a refuge.

We consider a particular case of the model (1), when $p_{13} = p_{31} = r$, $p_{11} = p_{22} = p_{33} = p_{44} = p$, $\beta = \gamma$, $\varepsilon = \delta$, model (1) takes the form:

$$\begin{aligned} \dot{x}_1 &= a_1 x_1 - p x_1^2 - r x_1 x_3 + \beta x_2 - \beta x_1, \\ \dot{x}_2 &= a_2 x_2 - p x_2^2 + \beta x_1 - \beta x_2, \\ \dot{x}_3 &= a_3 x_3 - p x_3^2 - r x_1 x_3 + \delta x_4 - \delta x_3, \\ \dot{x}_4 &= a_4 x_4 - p x_4^2 + \delta x_3 - \delta x_4. \end{aligned} \tag{2}$$

Analysis of models (1), (2) involves finding of stationary states corresponding to stationary population densities. The search for most stationary states for models (1),(2) in an analytical form is difficult due to the dimension of the models and a large number of parameters. However, in this situation, it is possible to search for particular sets of model parameters using evolutionary numerical optimization algorithms. Note that for a particular case of model (1), when $p_{13} = p_{31} = r$, $p_{11} = p_{22} = p_{33} = p_{44} = p$, $a_1 = a_2 = a_3 = a_4 = a$, $\beta = \gamma$, $\varepsilon = \delta$, the problem of finding parameters that ensure the mode of population coexistence is studied in [21].

3. Analysis of stationary states and construction of projections of phase portraits for a four-dimensional population-migration model (2)

In this section of the paper, we develop the approach proposed in [35] to the optimization search for the parameters of population models. The approach

considered in this paper makes it possible to obtain such sets of model parameters that ensure the achievement of the extremum of the quality criterion. We propose to use a quality criterion by analogy with [35] to study model (2).

We consider the optimization problem of species coevolution in system (2). Within the frames of this problem, we will track the change in the coefficients $a_1, a_2, a_3, a_4, p, r, \beta, \delta$. These coefficients are limited by some positive values contained in the parametric set A of system (2). Namely, one can write

$$(a_1, a_2, a_3, a_4, p, r, \beta, \delta) \in A. \tag{3}$$

One of the variants for the optimality conditions has the form

$$\int_0^{t_1} x_i(t)dt \to \max, \quad i = 1, 2, 3, 4, \tag{4}$$

where t_1 is the boundary of the studied time interval. However, in model (2), to take into account the requirement for the coexistence of two populations in three areas (in an area with interspecific competition and in two refuges) we will use instead of condition (4) an optimality condition of the form

$$\int_{t_0}^{t_1} x_1(t)x_2(t)x_3(t)x_4(t)dt \to \max, \tag{5}$$

where t_0 is the expected time for system (2) to reach the stationary mode. In the case of extinction of one of the two species (both in an area with interspecific competition and in two refuges), the integrand vanishes.

To solve the optimization problem (2),(5), we use the differential evolution algorithm. For the convenience of computational procedures, we write condition (5) in the form

$$\int_{t_0}^{t_1} (x_1(t)x_2(t)x_3(t)x_4(t)dt)^{-1} \to \min, \tag{6}$$

Estimation of fulfillment of condition (6) is the objective function (loss function) for the optimization algorithm. If this condition is met, then the algorithm terminates. Algorithm has the form:

```
def fit_evo(w):
        plane,time=eval_eco_4d(w)
        p = np.array(plane)
        p1 = p[:,0]; p2 = p[:,1];
        p3 = p[:,2]; p4 = p[:,3]
        px = p1*p2*p3*p4
        error = np.trapz(px[-30:])
        if error <= 0: return 1000
        else: return error**(-1)
```

For stable operation of the algorithm, negative and zero results are eliminated by assigning a fixed value of 1000. Optimization is carried out using

the differential evolution method implemented in the scientific computing library SciPy of Python language. The connection of differential evolution is carried out as follows:

```
scipy.optimize.differential_evolution(parametres).
```

With the help of a program created in the Python language, a computational experiment is carried out to select the parameters of the model (2). The parameter values are limited by the interval [0.3, 10], the maximum number of iterations of the differential evolution method is 100. The code snippet has the form:

```
bounds = list([(0.3,10) for i in range(8)])
x0 = list([0 for i in range(8)])
res = differential_evolution(fit_evo, bounds,
maxiter = 100, workers = 1, disp = True)
print(res)
```

The RK45 module (Runge–Kutta method) from SciPy is connected as a solver. Model (2) is defined by the following class:

```
class model_eco_4d(ode_model):
        def eq(self, t, y):
                [a1,a2,a3,a4], p, r, beta, delta =
                map(self.args.get, "a p r beta delta".split())
                x0,x1,x2,x3 = y
                dx1=a1*x0-p*x0**2-r*x0*x2+beta*x1-beta*x0
                dx2=a2*x1-p*x1**2+beta*x0-beta*x1
                dx3=a3*x2-p*x2**2-r*x0*x2+delta*x3-delta*x2
                dx4=a4*x3-p*x3**2+delta*x2-delta*x3
                return [dx1,dx2,dx3,dx4]
        def regulate(self, P):
                pass
        @property
        def current_time(self): return self.states.t
```

The solving of the optimization problem of searching for parameters of model (2), taking into account condition (6), made it possible to conduct a computational experiment and find the corresponding set of parameters under the following initial conditions $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$. The specified set of parameters has the form

$$(a_1, a_2, a_3, a_4, p, r, \beta, \delta) = (5.674e+00, 8.693e+00, 9.045e+00, 9.637e+00, 3.595e-01, 7.522e-01, 3.252e+00, 7.500e-01). \quad (7)$$

Taking into account rounding to two decimal places, the set of parameters will take the form: 5.67, 8.70, 9.05, 9.64, 0.36, 0.75, 3.25, 0.75. Using the found set of parameters, a positive stationary state is obtained: $x_1 = 5.09$, $x_2 = 17.73$, $x_3 = 15.84$, $x_4 = 25.99$. The considered approach to the search for stationary states is of interest due to the fact that for many high-dimensional population systems (including four-dimensional systems (1)–(2)) finding stationary states in a general form causes difficulties even when using computing packages.

Next, we consider examples of the construction of phase portraits projections for model (2). The projection of the phase portrait on the plane (x_1, x_2),

taking into account $x_3 = 15.84$, $x_4 = 25.99$, is shown in the figure 1. The projection of the phase portrait in space (x_1, x_2, x_4) with regard to $x_3 = 15.84$ for model (2) is shown in the figure 2.

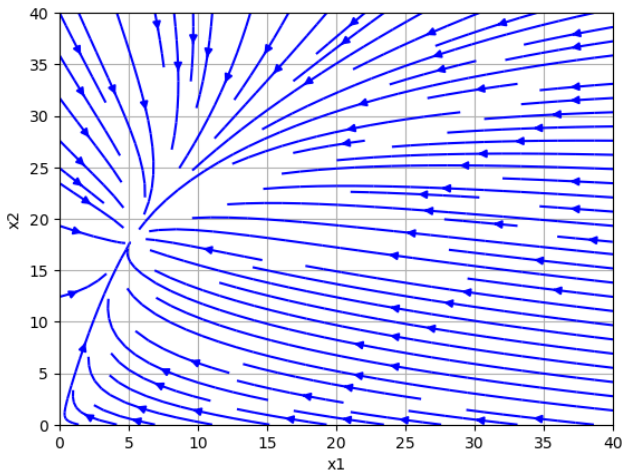

Figure 1. Projection of the phase portrait on the plane (x_1, x_2) for system (2) at $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$, $a_1 = 5.67$, $a_2 = 8.70$, $a_3 = 9.05$, $a_4 = 9.64$, $p = 0.36$, $r = 0.75$, $\beta = 3.25$, $\delta = 0.75$

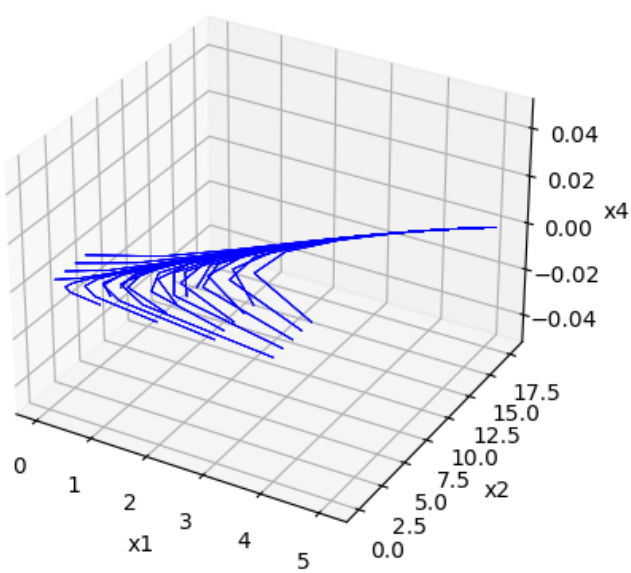

Figure 2. Projection of the phase portrait in space (x_1, x_2, x_4) for system (2) at $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$, $a_1 = 5.67$, $a_2 = 8.70$, $a_3 = 9.05$, $a_4 = 9.64$, $p = 0.36$, $r = 0.75$, $\beta = 3.25$, $\delta = 0.75$

The projection of the phase portrait on the plane (x_1, x_3), taking into account $x_2 = 17.73$, $x_4 = 25.99$, is shown in figure 3. The projection of the phase portrait in space (x_1, x_3, x_4) with regard to $x_2 = 17.73$ for model (2) is shown in the figure 4.

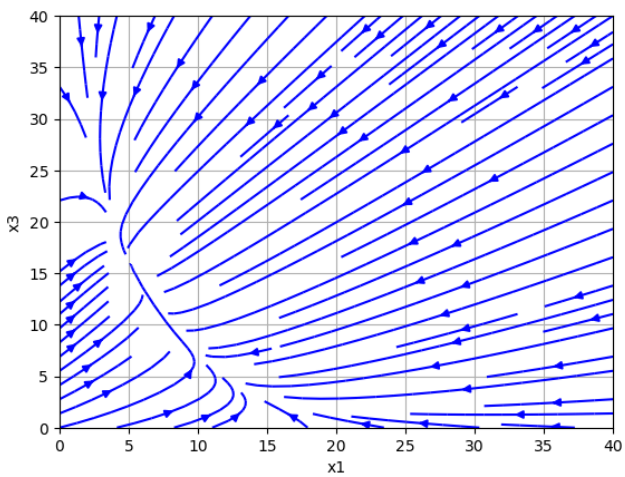

Figure 3. Projection of the phase portrait on the plane (x_1, x_3) for system (2) at $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$, $a_1 = 5.67$, $a_2 = 8.70$, $a_3 = 9.05$, $a_4 = 9.64$, $p = 0.36$, $r = 0.75$, $\beta = 3.25$, $\delta = 0.75$

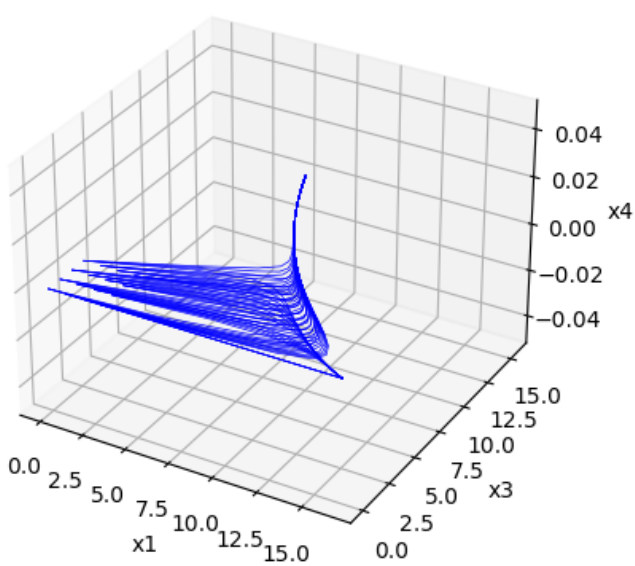

Figure 4. Projection of the phase portrait in space (x_1, x_3, x_4) for system (2) at $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$, $a_1 = 5.67$, $a_2 = 8.70$, $a_3 = 9.05$, $a_4 = 9.64$, $p = 0.36$, $r = 0.75$, $\beta = 3.25$, $\delta = 0.75$

The projection of the phase portrait on the plane (x_2, x_3), taking into account $x_1 = 17.73$, $x_4 = 25.99$, is shown in figure 5. The projection of the phase portrait in space (x_2, x_3, x_4), taking into account $x_2 = 17.73$ for model (2), is shown in the figure 6.

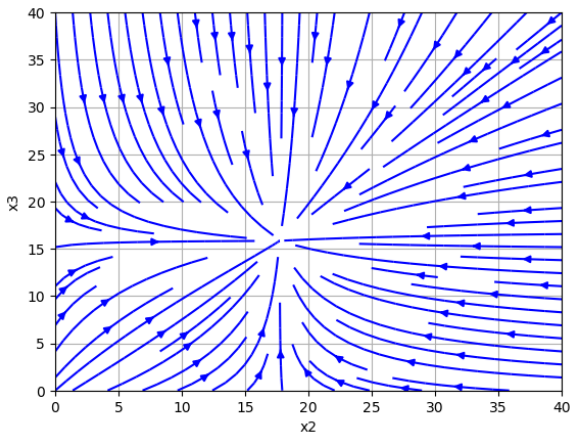

Figure 5. Projection of the phase portrait on the plane (x_2, x_3) for system (2) at $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$, $a_1 = 5.67$, $a_2 = 8.70$, $a_3 = 9.05$, $a_4 = 9.64$, $p = 0.36$, $r = 0.75$, $\beta = 3.25$, $\delta = 0.75$

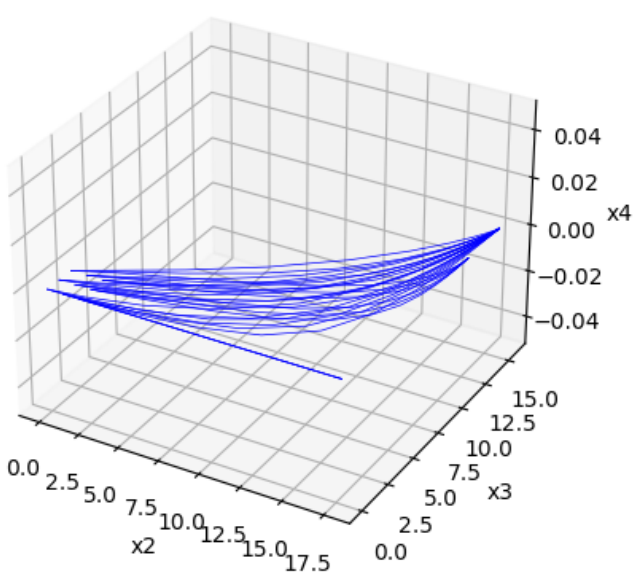

Figure 6. Projection of the phase portrait in space (x_2, x_3, x_4) for system (2) at $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$, $a_1 = 5.67$, $a_2 = 8.70$, $a_3 = 9.05$, $a_4 = 9.64$, $p = 0.36$, $r = 0.75$, $\beta = 3.25$, $\delta = 0.75$

According to figures 1–6 the stationary state $x_1 = 5.09$, $x_2 = 17.73$, $x_3 = 15.84$, $x_4 = 25.99$ has the character of a stable node. In this section, for clarity, separate variants for two-dimensional and three-dimensional projections are given taking into account the fact that a complete set of projections can be built in a similar way. The construction of phase portraits projections made it possible to analyze the nature of the dependence of phase variables on each other.

4. Construction of stochastic models "two competitors – two migration areas"

For further analysis of the qualitative properties of the models, we study the influence of stochastics introduction. For this purpose, it is proposed to develop a stochastic model by constructing self-consistent stochastic models [13, 30, 32, 43, 44]. This method involves, in the course of mathematical transformations, obtaining the coefficients of the Fokker–Planck equation from the interaction scheme, which allows implementing a symbolic record of all possible interactions in the system. The basic steps of this method are presented in the description of algorithm 1.

Algorithm 1: Obtaining a model from a system description

Initial parameters: description of the system.

Result: Deterministic and stochastic models (ODE and SDE systems).

1 **begin**

2 1. Adding interactions to the model.

3 2. Construction of the interaction scheme of the model.

4 3. Construction of system state operators from the interaction scheme (M and N).

5 4. Construction of an operator for changing the state of the system (R).

6 5. Construction of transition intensities (s).

7 6. Construction of the coefficients for the Fokker–Planck equation (A and B).

8 7. Construction of the SDE.

This algorithm is implemented in the Python programming language using the NumPy and SciPy libraries and it is described in [31]. The software package used in this paper makes it possible to construct a stochastic model of a dynamic system from its description, to construct the corresponding deterministic model, to obtain numerical solutions of ODEs and SDEs, as well as a graphical representation of the solutions.

When implementing algorithm 1, the input data is not an interaction scheme, but a description of the interactions occurring in the system. For this, the `PopModel` class is used. Calling the `PopModel(n)` constructor, where `n` is the dimension of the system, creates a class object that is a formal model of the system. The resulting object contains a description of all

considered system interactions. The `adder()` method of the `PopModel` class allows to add the main types of population interactions to the model, such as natural reproduction, natural death, competition, symbiosis, predator-prey relationships.

For the generalized model (1), we define a class object:

```
model_1 = pm.PopModel(4)
```

and add a description of interactions

```
# natural reproduction
model_1.adder(1,1,0,"a_1")
model_1.adder(1,2,0,"a_2")
model_1.adder(1,3,0,"a_3")
model_1.adder(1,4,0,"a_4")

# Intraspecific competition
model_1.adder(5,1,0,"p_11")
model_1.adder(5,2,0,"p_22")
model_1.adder(5,3,0,"p_33")
model_1.adder(5,4,0,"p_44")

# Interspecies competition
model_1.adder(6,3,1,"p_13")
model_1.adder(6,1,3,"p_31")

# Migration
model_1.adder(7,1,2,"γ")
model_1.adder(7,2,1,"β")
model_1.adder(7,3,4,"∈")
model_1.adder(7,4,3,"δ")
```

The first four lines of interactions description correspond to natural reproduction, lines 5–8 describe intraspecific competition, lines 9 and 10 are responsible for interspecific competition, and lines 11–14 describe population migration.

The `display_infos` method allows to display the interaction diagram. When using the Jupiter interactive shell, the output will have the form shown in the figure 7.

The IstoDE.py module allows to get the coefficients of the Fokker–Planck equation from the interaction diagram. The basic functions of this module are the `drift_vector` and `diffusion_matrix` functions. The first function is designed to obtain the drift vector A in the Fokker–Planck equation. With the help of the second function, we can get the diffusion matrix in the Fokker–Planck equation. As input parameters, these two functions take the state vector of the system, coefficients, matrices N and M.

In the figure 8 shows the result of the derivation of functions for obtaining the coefficients of the Fokker–Planck equation in relation to the interaction scheme in the figure 7.

Model (2) can be obtained from the constructed model by redefining the interaction coefficients as follows: $p_{13} = p_{31} = r$, $p_{11} = p_{22} = p_{33} = p_{44} = p$, $\beta = \gamma$, $\varepsilon = \delta$. In figure 9 shows the output of the `drift_vector` and `diffusion_matrix` functions for model (2).

Ввод [13]:
```
model_1.display_infos(model_1,XX)
```

$$x_1 = [a_1] \Rightarrow 2x_1$$
$$x_2 = [a_2] \Rightarrow 2x_2$$
$$x_3 = [a_3] \Rightarrow 2x_3$$
$$x_4 = [a_4] \Rightarrow 2x_4$$
$$2x_1 = [p_{11}] \Rightarrow x_1$$
$$2x_2 = [p_{22}] \Rightarrow x_2$$
$$2x_3 = [p_{33}] \Rightarrow x_3$$
$$2x_4 = [p_{44}] \Rightarrow x_4$$
$$x_1 + x_3 = [p_{13}] \Rightarrow x_3$$
$$x_1 + x_3 = [p_{31}] \Rightarrow x_1$$
$$x_1 = [\gamma] \Rightarrow x_2$$
$$x_2 = [\beta] \Rightarrow x_1$$
$$x_3 = [\epsilon] \Rightarrow x_4$$
$$x_4 = [\delta] \Rightarrow x_3$$

Figure 7. The output of the `display_infos` method

Ввод [9]:
```
f = de.drift_vector(XX, k_plus, model_1.matr_N(), model_1.matr_M())
sp.Matrix(f)
```

Out[9]:
$$\begin{bmatrix} a_1x_1 - p_{11}x_1^2 - p_{13}x_1x_3 - x_1\gamma + x_2\beta \\ a_2x_2 - p_{22}x_2^2 + x_1\gamma - x_2\beta \\ a_3x_3 - p_{31}x_1x_3 - p_{33}x_3^2 - x_3\epsilon + x_4\delta \\ a_4x_4 - p_{44}x_4^2 + x_3\epsilon - x_4\delta \end{bmatrix}$$

Ввод [10]:
```
g=de.diffusion_matrix(XX, k_plus, model_1.matr_N(), model_1.matr_M())
sp.Matrix(g)
```

Out[10]:
$$\begin{bmatrix} a_1x_1 + p_{11}x_1^2 + p_{13}x_1x_3 + x_1\gamma + x_2\beta & -x_1\gamma - x_2\beta & 0 & 0 \\ -x_1\gamma - x_2\beta & a_2x_2 + p_{22}x_2^2 + x_1\gamma + x_2\beta & 0 & 0 \\ 0 & 0 & a_3x_3 + p_{31}x_1x_3 + p_{33}x_3^2 + x_3\epsilon + x_4\delta & -x_3\epsilon - x_4\delta \\ 0 & 0 & -x_3\epsilon - x_4\delta & a_4x_4 + p_{44}x_4^2 + x_3\epsilon + x_4\delta \end{bmatrix}$$

Figure 8. The output of `drift_vector` and `diffusion_matrix` functions for model (1)

Ввод [20]:
```
f_3 = de.drift_vector(XX, k_plus_3, model_3.matr_N(), model_3.matr_M())
sp.Matrix(f_3)
```

Out[20]:
$$\begin{bmatrix} a_1x_1 - px_1^2 - rx_1x_3 - x_1\beta + x_2\beta \\ a_2x_2 - px_2^2 + x_1\beta - x_2\beta \\ a_3x_3 - px_3^2 - rx_1x_3 - x_3\delta + x_4\delta \\ a_4x_4 - px_4^2 + x_3\delta - x_4\delta \end{bmatrix}$$

Ввод [21]:
```
g_3=de.diffusion_matrix(XX, k_plus_3, model_3.matr_N(), model_3.matr_M())
g_3
```

Out[21]:
$$\begin{bmatrix} a_1x_1 + px_1^2 + rx_1x_3 + x_1\beta + x_2\beta & -x_1\beta - x_2\beta & 0 & 0 \\ -x_1\beta - x_2\beta & a_2x_2 + px_2^2 + x_1\beta + x_2\beta & 0 & 0 \\ 0 & 0 & a_3x_3 + px_3^2 + rx_1x_3 + x_3\delta + x_4\delta & -x_3\delta - x_4\delta \\ 0 & 0 & -x_3\delta - x_4\delta & a_4x_4 + px_4^2 + x_3\delta + x_4\delta \end{bmatrix}$$

Figure 9. The output of `drift_vector` and `diffusion_matrix` functions for model (2)

According to figures 8, 9, the drift vectors are fully consistent with the right parts of the systems of equations for models (1) and (2), respectively. This circumstance indicates that the drift vector can be used to study the deterministic behavior of the system. The result of the work of the `IstoDE.py` module of the software package is the construction of a stochastic and deterministic model of population dynamics for further study of trajectories.

5. Results of computer experiments

In this paper, for the numerical solution of systems of ordinary differential equations, a software implementation of the standard Runge–Kutta methods of fourth order is used. To solve the corresponding stochastic differential equations, we use a specially developed library, a detailed description of which is contained in [30, 31].

The results of numerical experiments, taking into account the parameters obtained in section 3, for the constructed deterministic model (2) and the corresponding stochastic model are shown in the figure 10. We consider the following initial conditions: $x_1(0) = 0.5$, $x_2(0) = 0.5$, $x_3(0) = 1$, $x_4(0) = 7$.

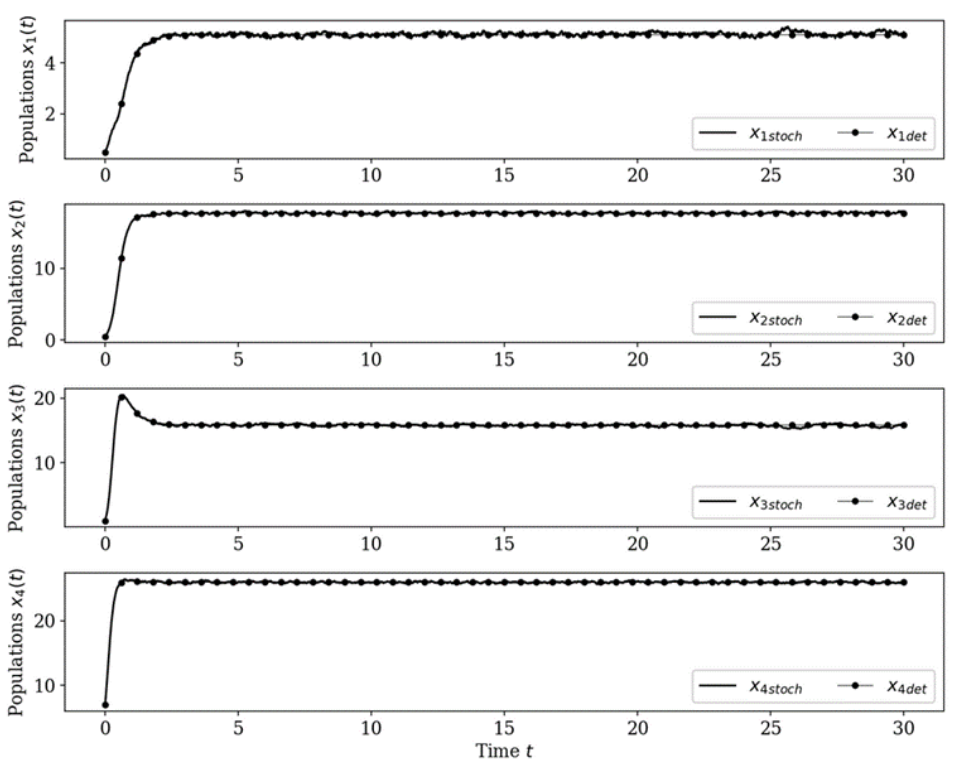

Figure 10. Visualization of the numerical solutions for model (2) and its corresponding stochastic model with parameters $a_1 = 5.67$, $a_2 = 8.70$, $a_3 = 9.05$, $a_4 = 9.64$, $p = 0.36$, $r = 0.75$, $\beta = 3.25$, $\delta = 0.75$ and with initial conditions $(x_1(0), x_2(0), x_3(0), x_4(0)) = (0.5, 0.5, 1, 7)$

According to figure 10, the introduction of stochastics has insignificant influence on the behavior of the model. As in the deterministic case, the trajectories of solutions of stochastic differential equations corresponding to model (2) are characterized by reaching the stationary mode.

6. Conclusion

The paper uses such an approach to the study of four-dimensional population-migration models, which is based on the implementation of evolutionary algorithms for searching for parameters, of the method of constructing self-consistent stochastic models and of modified methods for the numerical solution of finite-dimensional differential systems. The solution of the optimization problem by the method of differential evolution made it possible to find the optimal parameters of the «two competitors – two migration areas» model with competition of two species in the basic area and with migration to two refuges. For this model, an approximate positive stationary state is found that corresponds to the obtained set of parameters.

In this paper, based on the methods used, it is possible to construct new stochastic models of population dynamics, taking into account competition and bidirectional migration. The implementation of the stochastization algorithm for model (2) according to the given interaction schemes made it possible to analyze the trajectory dynamics of the stochastic model in comparison with the deterministic model.

As directions for further research, one can indicate the construction of new modifications of population-migration models based on model (1), the transition to the description and study of controlled ecological systems, as well as the identification of such sets of parameters that lead to a significant difference in the dynamics deterministic and stochastic models.

References

[1] V. Volterra, "Fluctuations in the abundance of a species considered mathematically," *Nature*, no. 118, pp. 558–560, 1926. DOI: 10.1038/118558a0.

[2] A. J. Lotka, *Elements of physical biology*. Baltimore, MD, USA: Williams and Wilkins Company, 1925.

[3] A. D. Bazykin, *Nonlinear dynamics of interacting populations [Nelineynaya dinamika vzaimodeystvuyushchikh populyatsiy]*. Moscow–Izhevsk: Institute of Computer Research, 2003, in Russian.

[4] A. Y. Aleksandrov, A. V. Platonov, V. N. Starkov, and N. A. Stepenko, *Study of mathematical modeling and sustainability of biological societies [Matematicheskoye modelirovaniye i issledovaniye ustoychivykh biologicheskikh soobshchestv]*. St. Petersburg: Lan, 2017, in Russian.

[5] P. Turchin, *Complex population dynamics*. Princeton: Princeton University Press, 2013.

[6] Y. A. Pykh, *Generalized Lotka–Volterra systems: theory and applications [Obobshchennyye sistemy Lotki–Vol'terra: teoriya i prilozheniya]*. St. Petersburg: SPbGIPSR, 2017, in Russian.

[7] L. Stucchi, J. Pastor, J. Garcia-Algarra, and J. Galeano, "A general model of population dynamics accounting for multiple kinds of interaction," *Complexity*, vol. 2020, p. 7 961 327, 2020. DOI: 10.1155/2020/7961327.

[8] J. S. Link, F. Pranovi, and S. Libralato, "Simulations and interpretations of cumulative trophic theory," *Ecological Modelling*, vol. 463, p. 109 800, 2022. DOI: 10.1016/j.ecolmodel.2021.109800.

[9] A. A. Shestakov, *Generalized direct method of Lyapunova for systems with distributed parameters [Obobshchennyy pryamoy metod Lyapunova dlya sistem s raspredelennymi parametrami]*. Moscow: URSS, 2007, in Russian.

[10] A. I. Moskalenko, *Methods of nonlinear mappings in optimal control. Theory and applications to models of natural systems [Metody nelineynykh otobrazheniy v optimal'nom upravlenii (teoriya i prilozheniya k modelyam prirodnykh sistem)]*. Novosibirsk: Nauka, 1983, in Russian.

[11] O. V. Druzhinina and O. N. Masina, *Methods for analyzing the stability of dynamic intelligent control systems [Metody analiza ustoychivosti dinamicheskikh sistem intellektnogo upravleniya]*. Moscow: URSS, 2016, in Russian.

[12] A. V. Demidova, O. V. Druzhinina, O. N. Masina, and A. A. Petrov, "Synthesis and computer study of population dynamics controlled models using methods of numerical optimization, stochastization and machine learning," *Mathematics*, vol. 9, no. 24, p. 3303, 2021. DOI: 10.3390/math9243303.

[13] A. V. Demidova, "Equations of population dynamics in the form of stochastic differential equations," *RUDN Journal of Mathematics, Information Sciences and Physics*, vol. 1, pp. 67–76, 2013, in Russian.

[14] A. V. Demidova, O. V. Druzhinina, and O. N. Masina, "Design and stability analysis of nondeterministic multidimensional populations dynamics models," *RUDN Journal of Mathematics, Information Sciences and Physics*, vol. 25, no. 4, pp. 363–372, 2017.

[15] I. N. Sinitsyn, O. V. Druzhinina, and O. N. Masina, "Analytical modeling and stability analysis of nonlinear broadband migration flows," *Nonlinear World*, vol. 16, no. 3, pp. 3–16, 2018, in Russian.

[16] Y. M. Svirezhev and D. O. Logofet, *Stability of biological communities*. Moscow: Nauka, 1978, in Russian.

[17] H. I. Freedman and B. Rai, "Can Mutualism alter Competitive Outcome: a Mathematical Analysis," *The Rocky Mountain Journal of Mathematics*, vol. 25, no. 1, pp. 217–230, 1995.

[18] Z. Lu and Y. Takeuchi, "Global asymptotic behavior in single-species discrete diffusion systems," *Journal of Mathematical Biology*, vol. 32, pp. 67–77, 1993. DOI: 10.1007/BF00160375.

[19] X.-a. Zhang and L. Chen, "The linear and nonlinear diffusion of the competitive Lotka–Volterra model," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 66, no. 12, pp. 2767–2776, 2007. DOI: 10.1016/j.na.2006.04.006.

[20] A. V. Demidova, O. V. Druzhinina, O. N. Masina, and E. D. Tarova, "Computer research of nonlinear stochastic models with migration flows," in *Proceedings of the Selected Papers of the 10th International Conference "Information and Telecommunication Technologies and Mathematical Modeling of High-Tech Systems" (ITTMM-2019). CEUR Workshop Proceedings*, vol. 2407, 2019, pp. 26–37.

[21] A. A. Petrov, O. V. Druzhinina, O. N. Masina, and I. I. Vasilyeva, "The construction and analysis of four-dimensional models of population dynamics taking into account migration flows," *Uchenye zapiski UlGU. Series: Mathematics and Information Technology*, no. 1, pp. 43–55, 2022, in Russian.

[22] I. I. Vasilyeva, "Computer modeling of the system of population dynamics taking into account the variation of migration parameters," *Uchenye zapiski UlGU. Series: Mathematics and Information Technology*, no. 2, pp. 21–30, 2022, in Russian.

[23] S. Cui and M. Bai, "Mathematical analysis of population migration and its effects to spread of epidemics," *Discrete and Continuous Dynamical Systems — B*, vol. 20, no. 9, pp. 2819–2858, 2015. DOI: 10.3934/dcdsb.2015.20.2819.

[24] H. C. Tuckwell, "A study of some diffusion models of population growth," *Theoretical Population Biology*, vol. 5, no. 3, pp. 345–357, 1974. DOI: 10.1016/0040-5809(74)90057-4.

[25] H. I. Freedman and P. Waltman, "Mathematical models of population interactions with dispersal. I: Stability of two habitats with and without a predator," *SIAM Journal on Applied Mathematics*, vol. 32, no. 3, pp. 631–648, 1977. DOI: 10.1137/0132052.

[26] L. J. S. Allen, "Persistence and extinction in single-species reaction-diffusion models," *Bulletin of Mathematical Biology*, vol. 45, no. 2, pp. 209–227, 1983. DOI: 10.1016/S0092-8240(83)80052-4.

[27] Y. Takeuchi, *Global dynamical properties of Lotka–Volterra systems*. Singapore: World Scientific, 1996.

[28] A. V. Demidova, O. V. Druzhinina, M. Jacimovic, O. N. Masina, and N. Mijajlovic, "Synthesis and analysis of multidimensional mathematical models of population dynamics," in *2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, vol. 2018-November, 2019, pp. 361–366. DOI: 10.1109/ICUMT.2018.8631252.

[29] A. Demidova, O. Druzhinina, M. Jacimovic, O. Masina, N. Mijajlovic, N. Olenev, and A. Petrov, "The Generalized algorithms of global parametric optimization and stochastization for dynamical models of interconnected populations," in *Optimization and Applications*, Cham: Springer International Publishing, 2020, pp. 40–54. DOI: 10.1007/978-3-030-62867-3_4.

[30] M. N. Gevorkyan, T. R. Velieva, A. V. Korolkova, D. S. Kulyabov, and L. A. Sevastyanov, "Stochastic Runge–Kutta software package for stochastic differential equations," in *Dependability Engineering and Complex Systems*, Cham: Springer International Publishing, 2016, pp. 169–179. DOI: 10.1007/978-3-319-39639-2_15.

[31] M. Gevorkyan, A. Demidova, T. Velieva, A. Korolkova, D. Kulyabov, and L. Sevastyanov, "Implementing a method for stochastization of one-step processes in a computer algebra system," *Programming and Computer Software*, vol. 44, pp. 86–93, Mar. 2018. DOI: 10.1134/S0361768818020044.

[32] A. Korolkova and D. Kulyabov, "One-step stochastization methods for open systems," *EPJ Web of Conferences*, vol. 226, p. 02 014, 2020. DOI: 10.1051/epjconf/202022602014.

[33] A. P. Karpenko, *Modern search engine optimization algorithms. Algorithms inspired by nature [Sovremennyye algoritmy poiskovoy optimizatsii. Algoritmy vdokhnovlennyye prirodoy]*, 2nd ed. Moscow: N.E. Bauman MSTU, 2016, in Russian.

[34] D. Simon, *Algorithms for evolutionary optimization [Algoritmy evolyutsionnoy optimizatsii]*. Moscow: DMK Press, 2020, in Russian.

[35] A. A. Petrov, O. V. Druzhinina, and O. N. Masina, "Application of the computational intelligence method to modeling the dynamics of multidimensional population system," in *Data Science and Algorithms in Systems*, vol. 597, Cham: Springer International Publishing, 2023, pp. 565–575. DOI: 10.1007/978-3-031-21438-7_45.

[36] R. Lamy, *Instant SymPy Starter*. Packt Publishing, 2013.

[37] T. E. Oliphant, "Python for scientific computing," *Computing in Science Engineering*, vol. 9, no. 3, pp. 10–20, 2007. DOI: 10.1109/MCSE.2007.58.

[38] C. Fuhrer, J. Solem, and O. Verdier, *Scientific computing with Python. Second edition*. Packt Publishing, 2021.

[39] C. Hill, *Learning scientific programming with Python*, Second Edition. Cambridge: Cambridge University Press, 2020.

[40] N. Sillero, J. C. Campos, S. Arenas-Castro, and A. M. Barbosa, "A curated list of R packages for ecological niche modelling," *Ecological Modelling*, vol. 476, p. 110 242, 2023. DOI: 10.1016/j.ecolmodel.2022.110242.

[41] O. V. Druzhinina, O. N. Masina, and E. D. Tarova, "Synthesis, computer research and stability analysis for the multidimensional models of the dynamics of the interconnected population," *Nonlinear World*, vol. 17, no. 2, pp. 48–58, 2019, in Russian. DOI: 10.18127/j20700970-201902-06.

[42] I. I. Vasilyeva, O. V. Druzhinina, and O. N. Masina, "Design and research of population dynamic model "two competitors – two migration areas"," *Nonlinear World*, vol. 20, no. 4, pp. 60–68, 2022, in Russian. DOI: 10.18127/j20700970-202204-06.

[43] C. W. Gardiner, *Handbook of stochastic methods: for Physics, Chemistry and the Natural Sciences*. Heidelberg: Springer, 1985.

[44] N. G. Van Kampen, *Stochastic processes in Physics and Chemistry*. Amsterdam: Elsevier, 1992.

For citation:

I. I. Vasilyeva, A. V. Demidova, O. V. Druzhinina, O. N. Masina, Construction, stochastization and computer study of dynamic population models "two competitors – two migration areas", Discrete and Continuous Models and Applied Computational Science 31 (1) (2023) 27–45. DOI: 10.22363/2658-4670-2023-31-1-27-45.

Information about the authors:

Vasilyeva, Irina I. — Assistant professor of Department of Mathematical Modeling, Computer Technologies and Information Security of Bunin Yelets State University (e-mail: `irinavsl@yandex.ru`, ORCID: https://orcid.org/0000-0002-4120-2595)

Demidova, Anastasia V. — Candidate of Physical and Mathematical Sciences, Assistant professor of Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University) (e-mail: `demidova-av@rudn.ru`, ORCID: https://orcid.org/0000-0003-1000-9650)

Druzhinina, Olga V. — Doctor of Physical and Mathematical Sciences, Chief Researche of Federal Research Center "Computer Science and Control" of Russian Academy of Sciences (e-mail: `ovdruzh@mail.ru`, ORCID: https://orcid.org/0000-0002-9242-9730)

Masina, Olga N. — Doctor of Physical and Mathematical Sciences, Deputy Head of Department of Mathematical Modeling, Computer Technologies and Information Security of Bunin Yelets State University (e-mail: `olga121@inbox.ru`, ORCID: https://orcid.org/0000-0002-0934-7217)

УДК 519.6:004.94
PACS 07.05.Tp, 02.60.Pn, 02.70.Bf
DOI: 10.22363/2658-4670-2023-31-1-27-45
EDN: VFNJWV

Построение, стохастизация и компьютерное исследование динамических популяционных моделей «два конкурента – два ареала миграции»

И. И. Васильева[1], **А. В. Демидова**[2], **О. В. Дружинина**[3], **О. Н. Масина**[1]

[1] *Елецкий государственный университет им. И. А. Бунина, ул. Коммунаров, д. 28, Елец, Липецкая обл., 399770, Россия*
[2] *Российский университет дружбы народов, ул. Миклухо-Маклая, д. 6, Москва, 117198, Россия*
[3] *Федеральный исследовательский центр «Информатика и управление» РАН, ул. Вавилова, д. 44, корп. 2, Москва, 119333, Россия*

Аннотация. При изучении детерминированных и стохастических популяционных моделей актуальными задачами являются формализация процессов с учётом новых эффектов, обусловленных взаимодействием видов, и развитие компьютерных методов исследования. Компьютерные методы исследования позволяют выполнить анализ траекторий многомерных популяционных систем. Мы рассматриваем модель «два конкурента – два ареала миграции», в которой учитывается внутривидовая и межвидовая конкуренция в двух популяциях, а также двунаправленная миграция обеих популяций. Для указанной модели мы учитываем вариативность параметров естественного воспроизводства видов. Предложено формализованное описание четырёхмерной модели «два конкурента – два ареала миграции» и её модификаций. С помощью реализации эволюционного алгоритма получен набор параметров, обеспечивающих сосуществование популяций в условиях конкуренции двух видов в основном ареале с учётом миграции этих видов. Исходя из полученного набора параметров найдено положительное состояние равновесия. Построены двумерные и трёхмерные проекции фазовых портретов. Осуществлена стохастизация модели «два конкурента – два ареала миграции» на основе метода построения самосогласованных одношаговых моделей. Для описания структуры модели использованы уравнения Фоккера–Планка. Выполнен переход к четырёхмерному стохастическому дифференциальному уравнению в форме Ланжевена. Для проведения численных экспериментов использован специализированный программный комплекс, предназначенный для построения и изучения стохастических моделей, а также разработана компьютерная программа на основе дифференциальной эволюции. Использованы алгоритмы генерирования траекторий винеровского процесса и многоточечных распределений и модификации метода Рунге–Кутты. В детерминированном и стохастическом случаях изучена динамика траекторий популяционно-миграционных систем. Проведён сравнительный анализ детерминированных и стохастических моделей. Результаты могут найти применение в задачах моделирования популяционных, экономических, демографических и химических систем.

Ключевые слова: модели динамики популяций, стохастические дифференциальные уравнения, одношаговые процессы, стохастизация, конкуренция, миграция, траекторная динамика, проекции фазовых портретов, компьютерное моделирование, программный комплекс

Discrete & Continuous Models
& Applied Computational Science
ISSN 2658-7149 (online), 2658-4670 (print)

2023, 31 (1) 46–63
http://journals.rudn.ru/miph

Research article

UDC 519.872:519.217
PACS 07.05.Tp, 02.60.Pn, 02.70.Bf
DOI: 10.22363/2658-4670-2023-31-1-46-63
EDN: VEMDSA

Causality relationship between foreign direct investments and economic improvement for developing economies: Russia case study

Kouame A. Brou[1], **Ivan V. Smirnov**[1, 2]

[1] *Peoples' Friendship University of Russia (RUDN University), 6, Miklukho-Maklaya St., Moscow, 117198, Russian Federation*
[2] *Federal Research Center "Computer Science and Control" of RAS, 44-2, Vavilova St., Moscow, 119333, Russian Federation*

(received: March 7, 2023; revised: March 20, 2023; accepted: April 10, 2023)

Abstract. Foreign direct investment (FDI) can have a significant impact on economic development in developing economies like Russia. FDI can bring in capital, technology, and management expertise that can stimulate economic growth, increase employment, and improve productivity. In the case of Russia, FDI has played a vital role in the country's economic development. A study conducted by the World Bank in 2019 found that FDI inflows have contributed significantly to Russia's economic growth and led to increased productivity, employment, and exports. The article analyzes the relationship between foreign direct investment and economic growth in Russia using ARDL cointegration and Toda–Yamamoto causality analysis test. The results reveal that there is no causality relation between GDP growth and foreign direct investment inflow in Russia. Overall, foreign direct investment effectively contributes to economic growth in Russia in the short term and not really in the long run.

Key words and phrases: foreign direct investment, economic growth, ARDL, Toda–Yamamoto causality

1. Introduction

Foreign direct investment (FDI) can play a significant role in economic development, particularly for developing economies. In the case of Russia, FDI has been seen as an important source of capital inflows and a means to improve the country's economic conditions. According to some economists, FDI contributes to the increase in the productive capacity of the economy and can also serve as a vector for the dissemination of technologies or knowledge [1, 2]. There is a causal relationship between foreign direct investment and economic improvement in Russia. FDI can bring in new technology, increase competition, create employment opportunities, and increase productivity in

the host country. Russia has been successful in attracting FDI in recent years, and this has contributed to the growth of various sectors of the economy, such as oil and gas, metals, and telecommunications. This brings up to date the debate on the effect of FDI on the economic growth of developing countries [3]. However, the relationship between FDI and economic development in Russia is not without challenges. One of the main challenges is the country's dependence on natural resources, particularly oil and gas. While FDI in the natural resource sector has contributed to the country's economic growth, it has also made the economy vulnerable to fluctuations in commodity prices. Some authors argue that FDI, i.e., investments carried out abroad by transnational or multinational companies with a view to acquire assets and manage production and marketing activities in host countries, positively affect economic growth [4].

Still others demonstrate that FDI only stimulates economic growth subject to the fulfillment of certain conditions, namely human capital, trade openness and good institutional governance [3, 5].

The objective of this article is therefore on the one hand to evaluate the relationship between FDI and economic growth in Russia, and on the other hand to highlight proposals from which the policies of economic improvement can rely on. The rest of the article is presented as follows: related works, material and methods, results and conclusion.

2. Related works

For years, extensive study has been conducted to determine the relationship between FDI entry into host nations and economic progress. The causal relationship between GDP growth and FDI, in theory, can go either way. One the one hand, FDI inflows can boost growth for the host countries through the expansion of the capital stock, the creation of new jobs, and the transfer of technology [6]. On the other hand, expanding economies draw new investment possibilities, including FDI inflows, to the host nation [7]. Despite the fact that more studies support the positive, a smaller number of studies indicated that domestic investment competition has a detrimental impact on economic growth.

Co-integration and panel Granger causality analyses in panel data was used to examine the connection between foreign direct investment and economic growth in 65 countries [8]. The findings reveal a discrepancy in the co-integration of the panel study's relationship. The findings also point to a one-way causal relationship between FDI and GDP, which may be useful in prioritizing the allocation of resources across sectors to encourage FDI.

The paper [9] explores the causal relationship between foreign direct investment and exports using annual data for 19 emerging economies in Asia from 1980 to 2015. China, Republic of Korea, Indonesia, Singapore, and Turkey have causality from export to FDI at 1% significant level, according to the first part of Granger Causality results. At a 5% level of significance, Nepal, Sri Lanka, the Philippines, Thailand, and Oman have a causal relationship between export and FDI. At a 10% level of significance, it is plausible to conclude that Bangladesh and India have a causal relationship between export and foreign direct investment, even though the likelihood value is extremely close to the 5% significance threshold. Sri Lanka, Indonesia, and Turkey have

a causality from FDI to export at a 1% significant level, while India, Nepal, and Thailand have a causality from FDI to export at a 5% significance level, according to the second portion of the Granger causality association tests. Eventually, at a 10% level of significance, Hong Kong, Bangladesh, Singapore, Bahrain, Oman, and Saudi Arabia were determined to have a causal association between FDI and export. In a nutshell, the export-led growth hypothesis holds true for Asian countries' growing economies.

Granger causality test based on the vector error correction model was used to investigate the causal relationship between the two variables throughout the time span 1980–2014 [10]. The empirical findings offer compelling evidence for FDI's causal role in Cambodia's economic growth (GDP). The study does not, however, support a direct causal relationship between GDP and FDI. The growth impact of FDI is thus properly supported in Cambodia, it can be inferred. The study [11] looks at the connection between trade, FDI, and economic growth in Greece from 1960 to 2002. There may be an equilibrium relationship over the long term, according to the cointegration study. The Granger causality test results demonstrated that there is a causal relationship between the variables under investigation. Under the open-door policy, economic growth, trade, and Investment seem to be mutually reinforcing.

The authors of [12] determine whether there is a causal link between foreign direct investments and economic growth for developing countries. The 30 developing nations with the highest GDP growth rates in 2016 are taken into account in this context. Additionally, Dumitrescu Hurlin panel causality analysis is used to examine annual data for these nations for the years 1991 through 2015. It has been determined that foreign direct investments and economic growth are related causally. In other words, it is acknowledged that FDI plays a significant role in driving economic expansion. This instance demonstrates how a country's economy might grow by luring foreign investors to make direct investments there. In the paper [13] the relationship between foreign direct investment (FDI) and the expansion of 117 nations throughout seven regions are investigated. The Granger causality approach and panel VAR/block exogeneity test were used to conduct predictive analysis among the panel series on a more recent panel dataset covering the years 2010–2020. In order to explore the interaction effects of the variables, which have not yet gained widespread acceptance in the field being examined, wavelet coherence techniques were also modified. The empirical findings show that FDI and economic growth both globally and in the Asian area are causally related in both directions. Contrarily, in the American region, the causality is unidirectional. For the majority of developed and emerging economies in the regional analysis, the results imply no causality.

The causal association between foreign direct investments and economic development in Togo from 1991 to 2009 was studied in [14]. They tested and established the causal link between FDI and Togo's economic growth using the Granger-causality. The study discovered a one-way link between FDI and GDP using time series data. It is possible to conclude that FDI causes GDP.

The relationship between foreign direct investment (FDI) and economic growth in the nations of the Organization of Eastern Caribbean States (OECS) is experimentally examined in [15]. The research estimates a dynamic panel growth model using the generalized method of moments employing panel data

consisting of annual data covering the period 1988–2013 from 34 countries, including the six OECS economies. The empirical findings indicate that while FDI has a beneficial impact on growth, on its own, it has very little of an effect. Its considerable impact is therefore primarily indirect. Moreover, infrastructure improvement and FDI interact favorably to boost economic growth, whereas FDI discourages local investment.

According to the analysis of previous literature, no study on Russia has yet been done on the causal relationship between foreign direct investment and economic improvement for developing economies.

3. Materials and methods

3.1. Data Description

The analysis makes use of Russian economic annual data from 1990 through 2020 from The World Development Indicator (WDI) provided by the World Bank. The level of GDP, inflows of foreign direct investment, population growth, inflation, government consumption, financial development, and investment are included in the considered statistics (see tables 1, 2).

Table 1

Definition of variables

Variables	Definition
GDPG	The growth rate of the GDP
FDI	Inflow of Foreign Direct Investment in percentage of GDP
GGFCE	General Government Final Consumption Expenditures
INF	The Consumer Price Index
POPG	The growth rate of the population
DCPS	Domestique Credit to Private Sector

Table 2

Descriptive and summary statistics

Variables	Mean	Standard Deviation	Minimum	Maximum
GDPG	0.736661	6.251199	-14.53107	10.00007
FDI	23.21095	1.494542	20.35158	25.69407
GGFCE	18.06510	1.715673	13.85744	21.067110
INF	109.4699	302.7925	2.878297	1481.166
POPG	-0.079966	0.228914	-0.460024	0.286681
DCPS	28.98768	19.67446	3.077914	59.96833

The most commonly used indicators for gauging economic success of a nation are its GDP and GDP per capita, which can be measured in terms of level or growth. Many metrics, including commonly used income statistics like GDP or GDP per capita, can be used to assess the economic success of a nation or region (measured either in level or growth terms). These metrics do have certain drawbacks, most notably the fact that they tend to overestimate national wealth and do not take into consideration overall welfare. Despite these problems, we employ per capita real GDP growth as the yardstick for measuring economic activity.

3.2. Methodology

We conducted an empirical research using cointegration and causal analysis to determine the relationship between foreign direct investments and economic growth in Russia. This method enables us to assess the impact of foreign direct investment on economic development over the long term as well as the short term.

In a single equation framework, auto regressive differentiated lag (ARDL) models are frequently used to investigate dynamic relationships with time series data. The differentiated lags element of the model allows the dependent variable's present value to depend on both its own historical realizations, or the autoregressive part, and the present and past values of other explanatory variables. Variables might be either stationary, non-stationary, or both. The ARDL model can be used to distinguish between long-term and short-term impacts, as well as to test for cointegration or, more broadly, the presence of a long-term relationship, in its portrayal of Error Correction (EC) term between the relevant variables. There will be answers to frequently asked questions and a step-by-step guide for doing the boundaries test to determine whether a long-term relationship exists [16]. This test is implemented as a post-estimate command that displays recently determined critical values for finite samples and approximative p-values.

To achieve our goals, we used a time series autoregressive distributed lag model (ARDL) as proposed by Pesaran, Shin, and Smith in their papers to estimate economic growth using a linear function that controls the interest variable, which is foreign direct investment.

3.2.1. Unit root tests

It is necessary to identify the order of integration of variables in any econometrics research. Verifying that the variables in the regression are either integrated of order zero I(0) or, at most, integrated of order one I(1) is essential for estimating an ARDL model. Each cross sectional series unit root test has an Augmented Dickey Fuller (ADF) regression as its default baseline:

$$\Delta y_t = y_{t-1} + \sum_{j=1}^{p} \phi_j \Delta y_{t-j} + \epsilon_t, \tag{1}$$

where $y = \rho - 1$.

The tests assess the null of unit root H_0: $y = 0$ ($\rho = 1$) against the alternative of stationarity H_1: $y < 0$ ($\rho < 1$).

3.2.2. The bounds test or cointegration test approach

Cointegration between series presupposes the existence of one or more long-term equilibrium relations between them, and these relations can be integrated with these series' short-term dynamics in an error-correction (vector) model that looks like this:

$$Y_t = AY_{t-1} + \sum_{i=1}^{p} B_i \Delta Y_{t-i} + U_t, \tag{2}$$

where Y_t is vector of stationary variables under study (whose dynamics are explained); B_i is matrix whose elements are parameters associated with ΔY_t; A is matrix of the same dimension as B_i (where $r(A)$ is the number of co-entering relations); Δ is the first difference operator.

To test the existence or not of cointegration between series, the econometric literature provides several tests or approaches including the test of Engle & Granger [17], those of Johansen [18] and Pesaran [16]. The cointegration test of Engle & Granger [17] only helps to verify the cointegration between two integrated series (1) of the same order (i.e. order of integration = 1), it is therefore adapted to the bivariate case and is thus proves to be less effective for multivariate cases. The cointegration test by Johansen was created for multivariate instances and allows the cointegration on more than two series to be verified. The Engle and Granger test's drawbacks for the multivariate case are addressed by the Pesaran test, which is based on vector error correction autoregressive modeling (VECM). However, this test also calls for all series or variables to be integrated in the same order, which is not always the case in actual use. So, we can utilize the cointegration test of Pesaran known as the "bounds test to cointegration" when we have several integrated variables of various orders (I(0), I(1)). If we use the Pesaran cointegration test to verify the existence of one or more cointegrating relationships between the variables in an ARDL model, we will say that we are using the "ARDL approach to cointegrating" or that we apply the test of cointegration by the staggered delays. The model that serves as the basis for the test of cointegration by staggered delays (test of Pesaran) is the following cointegrated ARDL specification (it takes the form of an error correction model or a VECM), when studying the dynamics between two series "X_t: and Y_t"

$$\Delta Y_t = \lambda_1 Y_{t-1} + \lambda_2 X_{t-1} + \sum_{i=1}^{p} \alpha_i \Delta Y_{t-i} + \sum_{j=0}^{q-1} b_j \Delta X_{t-j} + \pi_0 + \pi_1 + e_t. \tag{3}$$

This specification presents the ARDL model, i.e. relation (3), in the form of an ECM or a VEC, which assumes the existence of cointegration relations between series. Relation (3) can also be written as follows:

$$\Delta Y_t = \pi_0 + \pi_1 + \sum_{i=1}^{p} \alpha_i \Delta Y_{t-i} + \sum_{j=0}^{q-1} b_j \Delta X_{t-j} + \Theta U_{t-1} + e_t, \tag{4}$$

where Θ is the error correction term, adjustment coefficient or restoring force. Based on relation (3), after estimation, we will conclude that there is

a cointegration relation between Y_t and X_t if and only if: $0 < |\hat{\Theta}| < 1$ and rejection $H_0 : \Theta = 0$ ($\hat{\Theta}$ is statistically significant). There are two steps to follow to apply the Pesaran cointegration test, namely:

(i) the determination of the optimal lag first (AIC, SIC) and

(ii) the use of Fisher's test to verify the hypotheses (Cfr relation (3)):

$H_0 : \lambda_1 = \lambda_1 = 0$: existence of a cointegration equation;

$H_0 : \lambda_1 \neq \lambda_1 \neq 0$: absence of cointegration equation.

The test process requires that the Fisher values produced be compared to the critical values (limits) that Pesaran et al. simulated for various scenarios and thresholds. It should be noted that for the critical values, the lower limit (1st set) relates to the variables I and the upper limit (2nd set) takes up the values for which the variables are integrated of order 1.

Thereby:

— if *Fisher* > *Upper bound*: cointegration exists;
— if *Fisher* < *Lower bound*: cointegration does not exist;
— if *Lower bound* < *Fisher* < *Upper bound*: no conclusion.

An error correction model can assist in confirming the existence or absence of cointegration between variables thanks to the method of Pesaran. Under the framework of our investigation, this model will assume the following shape:

$$\Delta \mathrm{GDPG}_t = \left(a_0 + \sum_{i=1}^{p} a_{1i} \Delta GDPG_{t-1} + \sum_{i=0}^{q} a_{2i} \Delta LOGFDI_{t-1} + \right.$$
$$+ \sum_{i=0}^{q} a_{3i} \Delta POPG_{t-1} + \sum_{i=0}^{q} a_{4i} \Delta INFL_{t-1} + \sum_{i=0}^{q} a_{5i} \Delta GGCE_{t-1} +$$
$$\left. + \sum_{i=0}^{q} a_{6i} \Delta GCPS_{t-1} + \sum_{i=0}^{p} a_{7i} \Delta GFCF_{t-1} + \Theta U_{t-1} + e_t \right). \quad (5)$$

Relations (1) and (2) will be the subject of estimates. But first of all, we will:

— Determine the degree of integration of the variables (stationarity test): Augmented Dickey–Fuller/ADF test.
— Test the possible existence of a cointegration relationship between variables: cointegration test of Pesaran or bound cointegration test.
— Test the causality between the variables under study: causality test in the sense of Granger, causality test in the sense of Toda and Yamamoto.

In addition, let us specify that the ARDL model is not applicable for integrated variables of order > to 1.

3.2.3. Autoregressive distributed lag model

This dynamic model allows for the inclusion of temporal influences (such as adjustment times, expectancies, etc.) in the explanation of a variable. In a dynamic model, a dependent variable's (Y_t) lagged values, the independent variables' (X_t) present values, and their time-lagged values all contribute to

its explanation (X_{t-1}). The forms are as follows:

$$Y_t = \phi + \sum_{i=1}^{p} a_i Y_{t-i} + \sum_{j=0}^{q} b_j X_{t-j} + e_t. \tag{6}$$

Note that b_0 reflects the short-term effect of X_t on Y_t. To calculate the long-term effect of X_t on Y_t (i.e., λ), starting from the following long-term or equilibrium relation: $Y_t = K + \lambda X_t + u$ and then:

$$\lambda = \frac{\sum_{j=0}^{q} b_j}{1 - \sum_{i=1}^{p} a_i}.$$

As with any dynamic model, the information criteria (AIC, SIC and HQ) will be used to determine the optimal shift (p^* or q^*); an optimal lag is one whose estimated model offers the minimum value of one of the stated criteria. These criteria are: that of Akaike (AIC), that of Schwarz (SIC) and that of Hannan and Quinn (HQ). Their values are calculated as follows:

$$\mathrm{AIC}(p) = \log|\hat{\Sigma}| + \frac{2}{T} n^2 p, \tag{7}$$

$$\mathrm{SIC}(p) = \log|\hat{\Sigma}| + \frac{\log T}{T} n^2 p, \tag{8}$$

$$\mathrm{AIC}(p) = \log|\hat{\Sigma}| + \frac{2\log T}{T} n^2 p. \tag{9}$$

Using the variables $\hat{\Sigma}$ variance-covariance matrix of estimated residuals, T: the number of observations, p: the offset or lag of the estimated model, and n, the number of regressors. Any of these dynamic models can be used to visualize both the short-term dynamics and the long-term impacts of one or more explanatory variables on a variable that needs to be explained. This can only happen if the studied time series are cointegrated, allowing for the estimate of an error correction/ECM model. Two series are really said to be "cointegrated" if they are integrated of the same order, while a series is said to be "integrated of order d" if it requires differentiation "d" times in order to become stationary.

A stationary series is stationary in mean and in variance, if its mean ($E(Y_t) = c$) remains invariant or constant over time and its variance does not increase over time ($\mathrm{Var}(Y_t) = \sigma$) the same for its covariances ($\mathrm{Cov}(Y_t - c)(Y_{t-p} - c) = y_t$). A few econometric issues, such as collinearity between explanatory variables (as in the DL model) and autocorrelation of errors (as in the AR model), make it difficult to estimate an ARDL model using Ordinary Least Squares (OLS). Robust estimating methods are typically employed. In our study, we aim to identify the effects of foreign direct investment (FDI: variable of interest) on GDP growth (GDPG: dependent variable), accounting for other crucial control variables such as population

growth (POPG), inflation (INFL), government consumption (GGFCE), and financial development (DCPS).

Thus, we suggest that the following function (linear functional form) will be estimated using an ARDL model: GDPG=f (LOGFDI, POPG, INFL, GGCE, DCPS). The ARDL representation of the function will be as follows if we want to capture both the immediate and long-term effects of the aforementioned explanatory variables on economic growth:

$$\Delta GDPG_t = \Bigg(a_0 + \sum_{i=1}^{p} a_{1i}\Delta GDPG_{t-1} + \sum_{i=0}^{q} a_{2i}\Delta LOGFDI_{t-1} + \\ + \sum_{i=0}^{q} a_{3i}\Delta POPG_{t-1} + \sum_{i=0}^{q} a_{4i}\Delta INFL_{t-1} + \sum_{i=0}^{q} a_{5i}\Delta GGCE_{t-1} + \\ + \sum_{i=0}^{q} a_{6i}\Delta GCPS_{t-1} + b_1 GDPG_{t-1} + b_2 LOGFDI_{t-1} + b_3 POPG_{t-1} + \\ + b_4 INFL_{t-1} + + b_5 GGCE_{t-1} + b_6 GCPS_{t-1} + e_t \Bigg), \quad (10)$$

where Δ is the first difference operator; a_0 is the constant; $a_1, a_2, \dots a_6$ are short-term effects; $b_1, b_2, \dots, b_6$ are long-term dynamics of the model; $e \sim id(0, \sigma)$ is an error term (white noise).

Like with any dynamic model, we will utilize parsimony (2) to estimate the ideal shifts (p, q) of the ARDL model using the information criteria (Akaike-AIC, Shwarz-SIC, and Hannan-Quin, Adjusted R-square). Designing an ARDL model as described above (relation 1) assumes that the variables have a cointegration connection, which even affects how these variables' short- and long-term coefficients are estimated. The Pesaran cointegration test can be applied in two steps:

(iii) determination of the optimal offset above all (AIC, SIC);

(iv) use the Fisher test to verify the hypotheses:

$H_0 : b_1 = b_2 \dots b_9 = 0$: existence of cointegration relation;

$H_1 : b_1 \neq b_2 \dots b_9 \neq 0$: absence of cointegration relation.

3.2.4. Causality test: tests Toda–Yamamoto approach

Some detractors, who situate the Granger causality test mostly on the passive side of traditional causality tests, praise the Granger causality test's efficacy in the sense of Toda & Yamamoto [19].

Remembering that the Granger test only applies to stationary (stationarized) series makes it imperative to run preliminary analyses of the series' cointegration or stationarity prior to confirming any causal relationships. Unit root tests are not always impartial and are less effective on small samples. Additionally, by continuing to transform the series by the first difference in order to achieve stationarization or cointegration, we lose information about the level of the series, which level information should not be suppressed since it is beneficial to understanding the dynamics of the model under study (series).

So, for small samples, the Johansen cointegration test is susceptible to some choice parameters that are likely to weaken it: lag or shift (risk of estimating an underparameterized VAR) and existence (absence) of deterministic trend in the VAR and/or the cointegration space (risk of loss in degree of freedom). When the hypothesis of no cointegration (H_0) is true, these characteristics frequently contribute to biases that cause the hypothesis to be rejected.

Due to this flaw in the cointegration results and the biased nature of unit root tests, the Granger causality test (random outcome) is less successful and non-sequential approaches to assess the causality between series are proposed by Toda & Yamamoto [19]. These two authors proposed to estimate a VAR in corrected level (over-parameterized), to serve as a basis for the causality test, under the hypothesis of a potential probable cointegration between series that they integrate. According to these authors, the preliminary tests of stationarity and cointegration (sequential Granger procedures) are of little importance for the economist who must instead worry about testing the theoretical restrictions (they secure the level information) (explicitly).

The following is the Granger causality test approach suggested by Toda and Yamamoto:

— find the maximum integration order of the series under study ($d_{\max}$) by resorting to stationarity tests;
— determine the optimal lag or offset of the VAR at the level under study (k) or autoregressive polynomial (AR) using the information criteria (AIC, SIC and HQ);
— estimate a VAR in increased level of order $p = k + d_{\max}$.

Concerning the estimation of the VAR in increased level, the stationarity conditions of the series will define the number of lags to add to the VAR. In fact, for stationary series in level, no lag is added to the VAR (standard test procedure); on the other hand, for series I(1), a delay will be added to the VAR, and so on. By way of illustration, if we want to test the causality between two series h_t and m_t in the sense of Toda and Yamamoto, we will have to estimate the increased VAR as follows:

$$\Delta h_t = \left(a_0 + \sum_{i=1}^{k} a_{1,i} h_{t-1} + \sum_{j=k}^{p=k+d_{\max}} a_{2,j} h_{t-1} + \right. \\ \left. + \sum_{i=1}^{k} a_{1,i} m_{t-1} + \sum_{j=k}^{p=k+d_{\max}} a_{2,j} m_{t-1} + u_{1,t} \right), \quad (11)$$

$$\Delta m_t = \left(b_0 + \sum_{i=1}^{k} b_{1,i} m_{t-1} + \sum_{j=k}^{p=k+d_{\max}} b_{2,j} m_{t-1} + \right. \\ \left. + \sum_{i=1}^{k} b_{1,i} h_{t-1} + \sum_{j=k}^{p=k+d_{\max}} b_{2,j} h_{t-1} + u_{2,t} \right). \quad (12)$$

Testing limits on the first k coefficients of such an enhanced or purposefully over-parameterized VAR will serve as the causality test, with all other

parameters set to zero (they reflect a probable cointegration between series in the VAR). The test is based on the Wald statistic W, which is distributed according to x^2 a with n degrees of freedom, where n is the number of restrictions. This statistic is independent of the order of integration of the series and their cointegration. Accordingly, in the sense of Toda and Yamamoto, the test hypotheses are as follows:

$H_0 : a_{1i} = 0 (x_c^2 < x_t^2)$, p is value $x^2 > 5\%$: m_t cause h_t at short term;

$H_0 : b_{1i} = 0 (x_c^2 < x_t^2)$, p is value $x^2 > 5\%$: m_t cause h_t at short term.

It will be ensured that the order of maximum integration does not exceed the optimal lag $d_{\max}$ of the AR polynomial of the VAR to apply this test.

4. Results

4.1. Unit root test

As it's crucial to make sure that the order of integration is either zero or one for ARDL modeling, the empirical analysis should begin with the execution of the unit root test (see the table 3). For the purpose of looking for indications of stationarity, the Augmented Dickey Fuller (ADF) unit root tests are performed. Overall, the findings show that the variables analyzed have integration orders of I(0) or I(1), therefore we may use an ARDL model to estimate our relationship. Nevertheless, bound cointegration tests will be used to evaluate the null hypothesis of cointegration.

Table 3

Unit root test results

Variables	Level	First difference	Conclusion
GDPG	−2.508997	−7.375739***	I(1)
FDI	−2.501577	−5.544038***	I(1)
GGFCE	−4.228480***		I(0)
INF	−1.467817	−8.258210***	I(1)
POPG	−3.030919**		I(0)
DCPS	0.356963	−3.097987**	I(1)

Note in the table 3: $*$; $**$; $***$ indicate statistical significance at the 10%, 5%, and 1% level.

4.2. Cointegration test

The ARDL model must be estimated previously for the Pesaran cointegration test (see the table 4). The critical values (which constitute boundaries) will be compared to the derived test statistic, Fisher's F-value, as follows:

— if *Fisher* > *Upper bound*: cointegration exists;
— if *Fisher* < *Lower bound*: cointegration does not exist;
— if *Lower bound* < *Fisher* < *Upper bound*: no conclusion.

Table 4

Cointegration test results

Variables	GDPG, LOGFDI, GGFCE, DCPS, INFL, POP	
F-STAT	4.966073	
	Born <	Born>
10%	2.08	3
5%	2.39	3.38
1%	3.06	4.15

The results of the cointegration test at the limits confirm the existence of a cointegration relationship between the series under study (the value of F-stat is > that of the upper limit), which gives the possibility of estimating the long term of GDPG, LOGFDI, GGFCE, DCPS, INFL, POPG.

4.2.1. ARDL (error correction form) estimation

After confirming that the five variables are not integrated of an order equal or greater than I(2) and that the series are co-integrated, the next step is to estimate the panel ARDL regression as specified in ECM equation. The suitable lag length is selected based on the AIC lag selection criteria. The table 5 presents the empirical results on public debt and economic growth nexus conditioned on other explanatory variables in Russia and for the full sample period, 1990–2020 (in the tables 5, 6 ∗; ∗∗; ∗ ∗ ∗ indicate statistical significance at the 10%, 5%, and 1% level).

Table 5

Error correction model estimation (long-run coefficients)

Variables	ARDL esttimations	
	Coefficient	Standard Error
LOGFDI	1.081203	0.034941
GGFCE	−1.826693***	0.004357
INFL	−0.025460	0.075845
POPG	−0.036743***	0.231617
DCPS	−2.933879	0.200808

With regard to the tests which help to diagnose the estimated ARDL model, we note the absence of autocorrelation of the errors, there is no heteroscedasticity and there is normality of the errors (see the table 7).

Table 6

Error correction model estimation (short-run coefficients)

Variables	ARDL esttimations	
	Coefficient	Standard Error
ECT(-1)	−3.281689**	0.375258
D(GDPG(-1))	0.646779**	0.179682
D(LOGFDI)	1.598383**	0.461876
D(LOGFDI(-1))	0.046155	0.552017
D(LOGFDI(-2))	3.599007***	0.600330
D(GGFCE)	−1.982127***	0.181506
D(GGFCE(-1))	1.144808***	0.369579
D(GGFCE(-2))	1.322160**	0.217360
D(DCPS)	0.755284***	0.138276
D(DCPS(-1))	−0.410554**	0.152367
D(DCPS(-2))	0.412383**	0.139994
D(INFL)	−0.172183***	0.018028
D(INFL(-1))	0.036264***	0.004688
D(INFL(-2))	0.005463**	0.001819
D(POPG)	37.91637***	7.129612
D(POPG(-1))	3.309460	7.920245
D(POPG(-2))	−54.70903***	9.940868

Table 7

Estimated ARDL model diagnostic test results

Test hypothesis	Tests	Value (probability)
Autocorrelation	Breusch–Godfrey	2.90 (0.23)
Heteroskedasticity	Breusch–Pagan–Godfrey	0.23 (0.99)
Heteroskedasticity	Arch-test	0.52 (0.22)
Normality	Jarque–Bera	1.03 (0.53)

The error correction coefficient (ECT) illustrates how quickly the dynamic model will change to restore equilibrium after a disturbance. This coefficient, which is −3.28 in the full panel ARDL regression, indicates that equilibrium is attained in about 0.3 years. This ECT coefficient, which is both highly

significant and negative, lends support to the idea that economic growth and the regressors have a consistent, long-term relationship. Government spending and population expansion have a substantial long-term impact on economic growth. The negative coefficients indicate that the GDP will grow less as the variable increases. Foreign direct investment is not a factor that can be used as macroeconomic tools to improve growth in the long term.

Foreign direct investment has a large and immediate impact on economic growth in the short run. The cumulative effect of FDI is 5.22, which indicates that, all other things being equal, a 1% increase in FDI will result in an increase in GDP growth of 0.00522%. An increase in foreign direct investment will help the economy in the short term because the effect is favorable. Government spending immediately has a negative impact on growth, but the overall short-term impact is good.

4.3. Causality

Many empirical studies that examine the two-way causation between foreign direct investment and economic growth as well as macroeconomic observational data and previous studies have produced conflicting findings. The Toda–Yamamoto test, which is performed in the last section of the analysis, is shown in the table 8.

The Toda–Yamamoto causality test is employed to determine the causal direction. The test compares the null hypothesis—that there is no causality to an alternative that suggests causality. With a 95% level of confidence, the findings show that there is no causal relationship between new GDP growth and foreign direct investment inflow in Russia. We can also assume that there is a bidirectional causal relationship between new GDP growth and population increase at a 90% confidence level. Also, we can observe that foreign direct investment contributes to inflation, contrary to popular belief.

5. Conclusions

In order to assess the impact of foreign direct investment in identifying causal variables of economic growth in Russia through macro-economic factors on the period 1990–2020, we used Russian data from the World Development Index of the World Bank. Overall, according to the study's findings, foreign direct investment positively impacts Russia's economy, even if only temporarily and not in the long run. The Autoregressive Distributed Lag (ARDL) modeling approach that we utilized was an intriguing tool for decision-makers to examine the factors that could support Russia's economic progress. In the long run, population expansion and government consumption have detrimental repercussions, according to our findings. The Toda–Yamamoto causality tests, on the other hand, show that there is no causal relationship between foreign direct investment and economic growth. We can also assume that there is a bidirectional causal relationship between new GDP growth and population increase at a 90% confidence level. Finally, we may observe a bidirectional causal relationship between domestic lending to the private sector and foreign direct investment.

Table 8

Toda–Yamamoto causality test

Null hypothesis	Chi-sq	p-value
LOGFDI does not Cause GDPG	0.403745	0.8172
GDPG does not Cause LOGFDI	2.528015	0.2825
INFL does not Cause GDPG	1.357337	0.5073
GDPG does not Cause INFL	1.357337	0.5073
POPG does not Cause GDPG	5.278680	0.0714
GDPG does not Cause POPG	7.580598	0.0226
DCPS does not Cause GDPG	4.108184	0.1282
GDPG does not Cause DCPS	4.358495	0.1131
GGFCE does not Cause GDPG	0.408538	0.8152
GDPG does not Cause GGFCE	0.248191	0.8833
INFL does not Cause LOGFDI	1.279775	0.5274
LOGFDI does not Cause INFL	11.14007	0.0038
POPG does not Cause LOGFDI	6.020798	0.0493
LOGFDI does not Cause POPG	3.874079	0.1441
DCPS does not Cause LOGFDI	17.08424	0.0002
LOGFDI does not Cause DCPS	5.618667	0.0602
LOGFDI does not Cause GGFCE	2.296316	0.3172
GGFCE does not Cause LOGFDI	1.842384	0.3980

Acknowledgments

This paper has been supported by the RUDN University Strategic Academic Leadership Program.

References

[1] M. Blomstrom, A. Kokko, and M. Zejan, "Multinational corporations and productivity convergence in Mexico," in *Foreign Direct Investment*. London: Palgrave Macmillan, 2000, pp. 134–159. DOI: 10.1057/9780230598614_9.

[2] M. Blomstrom and E. Wolff, "Multinational corporations and productivity convergence in Mexico," in *Convergence of productivity: cross-national studies and historical evidence*. Oxford: Oxford University Press, 1994, pp. 133–138.

[3] R. Ekodo, M. Ndam, and K. Ousmanou, "Investissement direct étranger et croissance économique en zone CEMAC: Le rôle du capital humain," *Repéres et Perspectives Économiques*, vol. 4, no. 2, pp. 1–22, 2020. DOI: 10.34874/IMIST.PRSM/RPE/21529.

[4] B. Bellon and R. Gouia, *Investissements directs étrangers et développement industriel méditerranéen*, French. Paris: Economica, 1998.

[5] S. Globerman and D. Shapiro, "Global foreign direct investment flows: the role of governance infrastructure," *World Development*, vol. 30, no. 11, pp. 1899–1919, 2002. DOI: 10.1016/S0305-750X(02)00110-9.

[6] E. Borensztein, J. D. Gregorio, and J.-W. Lee, "How does foreign direct investment affect economic growth," *Journal of International Economics*, vol. 45, pp. 115–135, 1998. DOI: 10.1016/S0022-1996(97)00033-0.

[7] D. Rodrik, "Institutions for high-quality growth: what they are and how to acquire them," in *International Monetary Fund's Conference on Second-Generation Reforms, Washington, DC, November 8–9, 1999*, ser. NBER working paper series, 2000, pp. 3–31. DOI: 10.3386/w7540.

[8] S. M. Abbes, B. Mostéfa, G. Seghir, and G. Y. Zakary, "Causal interactions between FDI, and economic growth: evidence from dynamic panel co-integration," *Procedia Economics and Finance*, vol. 23, pp. 276–290, 2015. DOI: 10.1016/S2212-5671(15)00541-9.

[9] Ü. Arslan, Y. S. Çeli̇köz, and A. E. Güzel, "Causality relationship between foreign direct investment and export: the case of ddeveloping economies of Asia," vol. 8, no. 4, 2018. DOI: 10.18488/journal.aefr.2018.84.537.551.

[10] S. Sothan, "Causality between foreign direct investment and economic growth for cambodia," *Cogent Economics & Finance*, vol. 5, no. 1, p. 1 277 860, 2017. DOI: 10.1080/23322039.2016.1277860.

[11] M. Dritsaki, C. Dritsaki, and A. Adamopoulos, "A causal relationship between trade, foreign direct investment and economic growth in Greece," *American Journal of Applied Sciences*, vol. 1, no. 3, pp. 230–235, 2004. DOI: 10.3844/ajassp.2004.230.235.

[12] Z. Adalı and S. Yüksel, "Causality relationship between foreign direct investments and economic improvement for developing economies," vol. 1, no. 2, pp. 109–118, 2017. DOI: 10.24954/MJECON.2017.6.

[13] S. Nupehewa, S. Liyanage, D. Polkotuwa, M. Thiyagarajah, R. Jayathilaka, and A. Lokeshwara, *More than just investment: causality analysis between foreign direct investment and economic growth*, pp. 210–220, Nov. 2022. DOI: 10.1371/journal.pone.0276621.

[14] A. K. Mawugnon and F. Qiang, "The Relationship between foreign direct investment and economic growth in Togo [1991–2009]," in *Proceedings of the 8th International Conference on Innovation and Management*, 2010, pp. 1269–1273.

[15] N. Mamingi and K. Martin, "Foreign direct investment and growth in developing countries: evidence from the countries of the Organisation of Eastern Caribbean States," vol. 124, pp. 79–98, 2018. DOI: 10.18356/e270b670-en.

[16] M. Pesaran, Y. Shin, and R. J. Smith, "Bound testing approaches to the analysis of level relationship," *Journal of Applied Econometrics*, vol. 16, no. 3, 2001. DOI: 10.1002/JAE.616.

[17] R. Engle and C. Granger, "Co-integration and error correction: representation, estimation, and testing," vol. 55, no. 2, pp. 251–276, 1987. DOI: 10.2307/1913236.

[18] S. Johansen, "Statistical analysis of cointegration vectors," pp. 231–254, 1988. DOI: 10.1016/0165-1889(88)90041-3.

[19] H. Y. Toda and T. Yamamoto, "Statistical inference in vector autoregressions with possibly integrated processes," *Journal of Econometrics*, vol. 66, pp. 225–250, 1995. DOI: 10.1016/0304-4076(94)01616-8.

For citation:

K. A. Brou, I. V. Smirnov, Causality relationship between foreign direct investments and economic improvement for developing economies: Russia case study, Discrete and Continuous Models and Applied Computational Science 31 (1) (2023) 46–63. DOI: 10.22363/2658-4670-2023-31-1-46-63.

Information about the authors:

Brou, Kouame A. — PhD student of Department Information Technology of Peoples' Friendship University of Russia (RUDN University) (e-mail: broureino@gmail.com, ORCID: https://orcid.org/0000-0003-1996-577X)

Smirnov, Ivan V. — Candidate of Physical and Mathematical Sciences, Assistant Professor of Department Information technology of Peoples' Friendship University of Russia (RUDN University); Head of department of Federal Research Center "Computer Science and Control" Russian Academy of Sciences (e-mail: ivs@isa.ru, phone: +7(499) 135-90-20, ORCID: https://orcid.org/0000-0003-4490-2017)

УДК 519.872:519.217
PACS 07.05.Tp, 02.60.Pn, 02.70.Bf
DOI: 10.22363/2658-4670-2023-31-1-46-63
EDN: VEMDSA

Причинно-следственная связь между прямыми иностранными инвестициями и экономическим ростом в развивающихся странах: российский опыт

К. А. Бру[1], **И. В. Смирнов**[1, 2]

[1] *Российский университет дружбы народов, ул. Миклухо-Маклая, д. 6, Москва, 117198, Россия*
[2] *Федеральный исследовательский центр «Информатика и управление» РАН, ул. Вавилова, д. 44, корп. 2, Москва, 119333, Россия*

Аннотация. Прямые иностранные инвестиции (ПИИ) могут оказать значительное влияние на экономическое развитие развивающихся стран, таких как Россия. ПИИ привлекают капитал, технологии и управленческий опыт, что может стимулировать экономический рост, увеличить занятость и повысить производительность. В случае России ПИИ сыграли жизненно важную роль в экономическом развитии страны. Исследование, проведенное Всемирным банком в 2019 году, показало, что приток ПИИ в значительной степени способствовал экономическому росту России и привел к повышению производительности, занятости и экспорта. В статье анализируется взаимосвязь между прямыми иностранными инвестициями и экономическим ростом в России с использованием метода коинтеграции ARDL и подхода к причинно-следственному анализу Тода–Ямамото. Результаты нашего анализа показывают отсутствие причинно-следственной связи между притоком прямых иностранных инвестиций в Россию и ростом ВВП. Однако в целом прямые иностранные инвестиции эффективно способствуют экономическому росту в России в краткосрочной перспективе.

Ключевые слова: прямые иностранные инвестиции, экономический рост, модель ARDL, причинно-следственная связь Тода–Ямамото

Discrete & Continuous Models
& Applied Computational Science
ISSN 2658-7149 (online), 2658-4670 (print)

2023, 31 (1) 64–74
http://journals.rudn.ru/miph

Research article

UDC 004.912
DOI: 10.22363/2658-4670-2023-31-1-64-74
EDN: VNWSXI

Methods of extracting biomedical information from patents and scientific publications (on the example of chemical compounds)

Nikolay A. Kolpakov[1],
Alexey I. Molodchenkov[2,3], **Anton V. Lukin**[2,3]

[1] *Moscow Institute of Physics and Technology (MIPT),*
9, Institutskiy Pereulok, Dolgoprudny, Moscow Region, 141700, Russian Federation
[2] *Federal research center "Computer science and control" of RAS,*
44-2, Vavilova St., Moscow, 119333, Russian Federation
[3] *Peoples' Friendship University of Russia (RUDN University),*
6, Miklukho-Maklaya St., Moscow, 117198, Russian Federation

(received: March 9, 2023; revised: March 23, 2023; accepted: April 10, 2023)

Abstract. This article proposes an algorithm for solving the problem of extracting information from biomedical patents and scientific publications. The introduced algorithm is based on machine learning methods. Experiments were carried out on patents from the USPTO database. Experiments have shown that the best extraction quality was achieved by a model based on BioBERT.

Key words and phrases: machine learning, natural language processing, named entity recognition, biomedical texts processing

1. Introduction

Every year the number of biomedical patents and scientific publications increases significantly. Often these texts don't contain any descriptive metadata, and this, in turn, leads to a large amount of unstructured data. Consequently, there is an increasing need for tools that could accurately extract the required information from such texts.

To extract infromation from texts for further processing, both machine learning approaches and algorithms based on regular expressions can be used. In [1, 2], regular expressions play a key role, and, on the contrary, in [3, 4], achievements in the field of deep machine learning, in particular the model of conditional random fields, are used. And in [5], a transformer-based machine learning technique is used, which, with proper parameter settings, can extract biomedical data quite well.

Although tools have been created for analyzing and interacting with unstructured data, these solutions are often based on rules that are applicable

to the specific data being processed. In this paper, we propose a solution to the problem of extracting biomedical information from patents using regular expressions. Thus, the resulting structured information can be used to train complex neural network models that will allow us to correctly extract information from a larger number of texts.

2. Related work

There aren't many solutions that solve the problem. Often, existing algorithms are designed to solve a large range of problems, so they they do not give sufficiently high results when solving the task of extracting definitions from biomedical patents and scientific publications.

For example, Jinhyuk Lee and his colleagues presented BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining) [5], a transformer language model [6] developed for automatic processing of the language of the biomedical field, which is pre-trained on large biomedical texts. This model can extract biomedical named entities, biomedical relationships in the text, and can also provide answers to biomedical questions. BioBERT is initialized with the values of weight functions that were obtained for BERT [7] (this model was previously trained on texts from the English Wiki and BooksCorpus), after which BioBERT was further trained on biomedical texts (this includes annotations from PubMed and full-text PMC articles).

The article [3] presents a different approach to solving NLP problems in the field of biomedicine. CLAMP (Clinical Language Annotation, Modeling, and Processing) uses both machine learning-based and rule-based methods to extract information. This toolkit allows you to extract named entities, split text into tokens, and much more. In their program, the authors use 3 types of tokenizers (to choose from):

1) OpenNLP tokenizer [8] based on machine learning;
2) tokenizer based on the separation of words by specified characters;
3) a rule-based tokenizer with various configuration parameters.

And for the task of extracting named entities, the authors suggest using:

1) conditional random fields algorithm (CRF) [9];
2) an algorithm based on a dictionary with a large amount of biomedical vocabulary collected from various resources, such as UMLS;
3) a regular expression-based algorithm for objects with common patterns.

OSCAR4 (Open-Source Chemistry Analysis Routines) [2] is an open system for automatic extraction of chemical terms from scientific articles. The basis of this work is the identification of chemicals based on regular expressions and identification based on a dictionary of predefined words. But to identify complex chemical compounds (which consist of several tokens), the Markov model of maximum entropy is used.

Also, there is a work [1] where the authors use morphology to extract biomedical words. The chemical object recognition system consists of two subsystems. The first extracts chemical objects and marks them in a normalized input document using a dictionary of predefined words and a morphological

approach. The morphology-based approach identifies the various elements in a chemical compound and combines them to create a final compound.

The second subsystem extracts additional chemical elements and distributes all recognized objects into classes of compounds and has such capabilities as decoding abbreviations and correcting spelling errors. In order to determine whether a certain entity is "chemical", the authors collected statistical information for each individual object. This information is used as the last stage of the extraction of named entities and is intended for the classification of the extracted object (either the object is chemical or not). These methods extract information from biomedical texts in general — they aren't aimed at extracting Markush structures [10] (see figure 1).

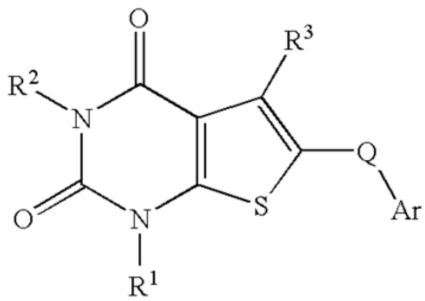

Figure 1. An example of a Markush structure, taken from US Patent 20040171623

3. The problem statement

Data concerning various biomedical patents are publicly available in various patent offices. Patents usually have a clear structure, which includes patent name, abstract, description, Claims and bibliographic information (date, patent number, authors).

The section we are interested in is Claims (see the figure 2), contains a description of the chemical compounds that are claimed by the authors of the patent. This is exactly the purpose of the legal protection provided by the patent. The Claims section may contain within itself several subsections that contain information on different chemical chains.

The connections presented in the Claims section can be described using the Markush structure [10] (see the figure 1). To find patents whose Markush structure is either the same or similar, you need to compare these structures. Since the Markush structure is a network model, then comparing such models directly is a very resource-intensive process. Therefore, so-called fingerprints are often used, which reflect the information presented in the Markush structures. But before that, you need to extract the information that is included in such structures, which is what this work is aimed at.

The task consists in extracting chemical compounds from the Claims section (see the table 1), names of variables (in place of which various values can be substituted), chemical elements, formulas and InChI codes [11] (see the figure 3) in order to transform this textual information into some structure of formal representation.

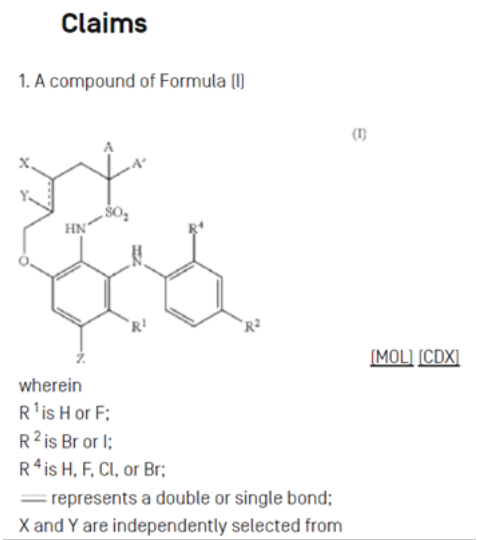

Figure 2. An example of the data in the Claims section, taken from US Patent 20120208859

Table 1

Examples of chemical compounds

Compound Name	Formula
nitrogen monoxide	NO
glucose	$C_6H_12O_6$
copper (II) sulfate	$CuSO_4$
carbon dioxide	CO_2
dichlorine heptoxide	Cl_2O_7

In the set-theoretic annotation, the problem can be formulated as follows: there are patents and scientific publications X where each element $x \in X$ is represented as $x = x_1, \dots, x_n$ ($x_1, \dots, x_n$ – is a sequence of words (tokens)), and a set of classes $Y = (y_1, \dots, y_5)$ is given, where:

— y_1 is the Claim number;
— y_2 is the variable to which we are looking for a description;
— y_3 is the description of the variable;
— y_4 is a link to another Claim;
— y_5 is in cases if the token doesn't match $y_1, \dots, y_4$.

It is necessary to construct a function F, that maps each element $x \in X$ to the corresponding element $y \in Y$.

The task of extracting information from texts is the search and classification of named entities (Named Entity Recognition) represented in unstructured text, according to predefined categories. A named entity is an n-gram in the text for which a category (class, label) is defined.

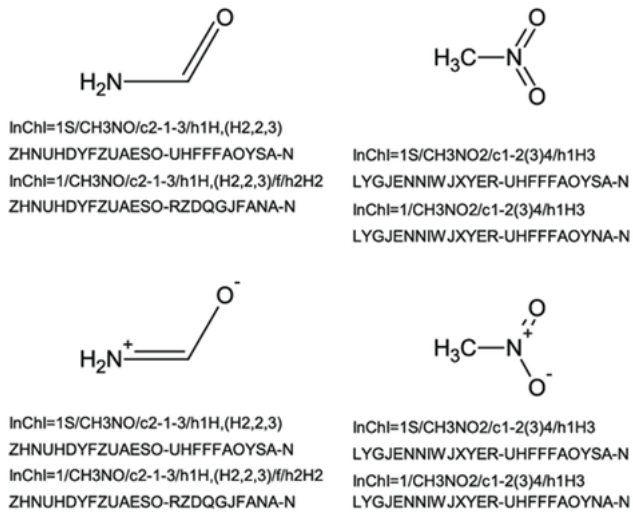

Figure 3. Examples of InChI codes [11]

4. Methodology

The algorithm for extracting information from texts can be divided into the following steps:

1) compilation of a dataset;
2) input data pre-processing;
3) data vectorization and feature extraction;
4) model training for extracting the necessary information from texts.

Compilation of the dataset also includes automated token markup. Data pre-processing includes normalization and tokenization of input data. The scheme of the algorithm is shown in figure 4.

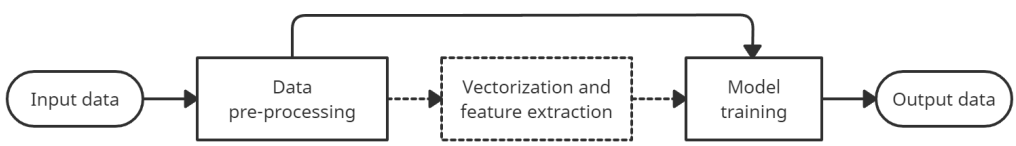

Figure 4. An algorithm for extracting information from texts

4.1. Compilation of a dataset

The data we worked with was taken from the USPTO patent database [12]. All data is initially presented in XML files that contain structured information about patents: description, annotation, bibliographic data, and Claims. To develop the algorithm, only the Claims section is taken from the files.

4.2. Input data pre-processing

The first stage of data processing is the extraction of the Claims section from the available dataset. Since such data has a similar design, the extraction is performed using regular expressions.

After extracting Claims, it is necessary to prepare the data for further work. To do this, the following string normalization is performed:

1. Extra spaces at the beginning and end of lines are removed.
2. Empty lines are also removed.
3. Each line is split in such a way that they contain only one description of variables. This is done by searching in each line of the following construction: *...variables ...definition-verb ...definitions ...definition-end-symbol*. At the same time, the situation is considered when a description of nested variables can be provided on the same line. For example, "Z is OR3, wherein R3 is C1-C6 alkyl". In this case, the string doesn't split.
4. If there is no *definition-end-symbol* in the string, then the strings are combined until the desired character is found.
5. If the line is the initial one for the Claim, but the Claim numbers are separated by a dash, then the subsequent content is copied for each Claim number from the given interval.

The resulting rows are then grouped by Claim. All the actions described above to normalize strings are also performed using regular expressions.

The use of normalization will allow us to train the model on a small sample more efficiently, and also will increase its accuracy. Grouping and splitting the rows will simplify the subsequent markup of the data.

The next step is to assign each token a label from the possible:

— CLAIM is Claim number;
— VAR is the variable that we are looking a description for; the description of this variable is substituted only to the last place where it was mentioned before the meeting of this variable;
— VAR-ALL is the variable that we are looking a description for; the description of this variable is substituted in all places where it was mentioned;
— DEF is description of the variable;
— REF is the link to another Claim;
— O is in case none of the above labels are assigned to the token.

The assignment of the corresponding label to tokens is carried out using the marking tools, provided by Federal research center "Computer science and control" of RAS. The result of these tools is data, containing the token, its label, the line number where it was found and the unique Claim number.

4.3. Vectorization and feature extraction

Since not all classification models accept string data values as input, it is necessary to vectorize such features. These include tokens and corresponding labels.

If a number can be associated with each unique label, then the situation is completely different with tokens. For each token, a vector of dimension 100 is constructed using the Word2Vec model [13, 14] to obtain vector representations of natural language words.

Word2Vec was trained on the collected dataset. The training took place for 10 epochs, with a sliding window size of 8.

To further train the machine learning model, it is necessary to combine tokens into lists based on Claims membership, and then submit these lists to Word2Vec as input. The result of such a model will be the mapping of each token to its vector representation.

Some machine learning algorithms, for example, based on Conditional Random Fields (CRF), are working better with features containing information about neighboring tokens relative to the one under consideration.

Therefore, another way of representation the data submitted to the input of such models is to match each token with a set of features. These features are:

— the token itself;
— the last 2–3 characters of the token;
— flag whether the token starts with a capital letter;
— flag whether the token is a number;
— flag whether the token contains only uppercase letters;
— information about neighboring tokens (a neighboring token and 3 flags, as in the previous points).

4.4. Model training

Both classical machine learning methods (Support Vector Machine [15], Conditional Random Fields [9]) and deep learning models (Stanford NER [16], BERT [7], BioBERT [5]), which are already pre-trained, were used as classification methods that would assign labels to previously unknown data based on marked and vectorized data.

Fine-tuning a pre-trained model, such as BioBERT, BERT and Stanford NER, was carried out on the data obtained in section 4.2 are tokens, labels, and Claims numbers. For BioBERT and BERT, in order to prevent overfitting, the number of epochs was chosen equal to 5 (see the figure 5).

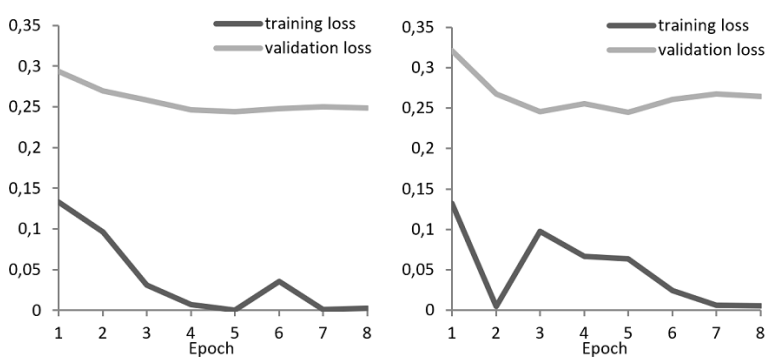

Figure 5. Epoch vs Loss graphs. Left graph is for BioBERT. Right graph is for BERT

The support vector machine (SVM) method was trained from scratch on vectorized data, and the conditional random field method (CRF) was trained on extracted features obtained in section 4.3.

5. Experiments

Within this work, a series of experiments was carried out to solve the classification problem. The experiments were conducted on 100 documents with more than 1,700 Claims. The training sample consisted of 70 documents, and the validation sample consisted of 30 documents.

Standard quality metrics were used to compare the results: precision, recall and F1-score [17]. Let's look at them in more detail.

To begin with, let's consider what TP, FP and FN are:

— TP is the number of tokens that the classifier has assigned the correct labels to;
— FP is the number of tokens that have the label O, but the classifier assigned them a different label;
— FN is the number of tokens that have a certain label (not O), but the classifier assigned them to another group.

Accuracy is the proportion of tokens that belong to a particular class, relative to all tokens that the classifier has assigned such a class label to. This metric is calculated:

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{1}$$

Recall is the proportion of tokens that the classifier has assigned a specific class label to, relative to all tokens that have this label. This metric is calculated by equation:

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{2}$$

F1-score is the average harmonic value of accuracy and recall. This metric is calculated by equation:

$$F1 - score = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 * TP}{2 * TP + FP + FN}. \tag{3}$$

The classification results based on test data are shown in the table 2.

Table 2

Metrics values for various classification methods

Model Name	Precision	Recall	F1-score
SVM	0.5276	0.6340	0.5675
CRF	0.6701	0.6358	0.6378
Stanford NER	0.7530	0.8488	0.7981
BERT	0.8437	0.8978	0.8699
BioBERT	0.8467	0.9012	0.8731

From the performed experiments, it can be seen that classical machine learning methods show results much worse than pre-trained deep learning models, which, in turn, classify tokens at a fairly good level.

6. Conclusion

This article describes a method for solving the problem of extracting information from biomedical texts for its further processing. This method makes it possible to extract a description of chemical compounds that are claimed by the authors of patents. The machine learning models, such as SVM, CRF, Stanford NER, BERT and BioBERT, on which experiments were afterwards carried out, were trained. In the future, it is planned to convert the received data into the InChI code format and write fingerprints that correspond to the Markush structures claimed by the patent authors. It is also planned to conduct another series of experiments to improve the quality of information extraction from texts.

References

[1] S. A. Akhondi *et al.*, "Automatic identification of relevant chemical compounds from patents," *Database: the journal of biological databases and curation*, vol. 1, pp. 1–14, 2019. DOI: 10.1093/database/baz001.

[2] D. Jessop, S. Adams, E. Willighagen, L. Hawizy, and P. Murray-Rust, "OSCAR4: A flexible architecture for chemical textmining," *Journal of cheminformatics*, vol. 3, no. 1, pp. 1–12, 2011. DOI: 10.1186/1758-2946-3-41.

[3] E. Soysal *et al.*, "CLAMP — a toolkit for efficiently building customized clinical natural language processing pipelines," *Journal of the American Medical Informatics Association*, vol. 25, no. 3, pp. 331–336, 2017. DOI: 10.1093/jamia/ocx132.

[4] M. Swain and J. Cole, "ChemDataExtractor: a toolkit for automated extraction of chemical information from the scientific literature," *Journal of Chemical Information and Modeling*, vol. 56, no. 10, pp. 1894–1904, 2016. DOI: 10.17863/CAM.10935.

[5] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. So, and J. Kang, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics (Oxford, England)*, vol. 36, no. 4, pp. 1234–1240, 2019. DOI: 10.1093/bioinformatics/btz682.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, pp. 4171–4186, 2018. DOI: 10.18653/v1/N19-1423.

[8] *The OpenNLP Project*, http://opennlp.apache.org, Accessed: 2023-03-07.

[9] *CRFsuite: a Fast Implementation of Conditional Random Fields (CRFs)*, http://www.chokkan.org/software/crfsuite/, Accessed: 2023-03-07.

[10] J. M. Bernard, "Handling of Markush Structures," *Journal of chemical information and computer sciences*, vol. 31, no. 1, pp. 64–68, 1991. DOI: 10.1021/ci00001a010.

[11] S. Heller, A. McNaught, I. Pletnev, S. Stein, and D. Tchekhovskoi, "The IUPAC International Chemical Identifier," *Journal of Cheminformatics*, vol. 7, pp. 1–34, 2015. DOI: 10.1186/s13321-015-0068-4.

[12] *USPTO*, https://www.uspto.gov/patents, Accessed: 2023-03-07.

[13] T. Mikolov, G. Corrado, K. Chen, and J. Dean, "Efficient estimation of word representations in vector space," *Proceedings of Workshop at ICLR*, pp. 1–12, 2013.

[14] T. Mikolov, W.-T. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," *Proceedings of NAACL-HLT*, pp. 746–751, 2013.

[15] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 03, pp. 273–297, 1995. DOI: 10.1007/BF00994018.

[16] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by Gibbs sampling," *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pp. 363–370, 2005. DOI: 10.3115/1219840.1219885.

[17] T. M. Mitchell, *Machine learning*. McGraw-Hill New York, 1997, 432 pp.

For citation:

N. A. Kolpakov, A. I. Molodchenkov, A. V. Lukin, Methods of extracting biomedical information from patents and scientific publications (on the example of chemical compounds), Discrete and Continuous Models and Applied Computational Science 31 (1) (2023) 64–74. DOI: 10.22363/2658-4670-2023-31-1-64-74.

Information about the authors:

Kolpakov, Nikolay A. — Master's degree student of Phystech School of Applied Mathematics and Informatics of Moscow Institute of Physics and Technology (e-mail: kolpakov.na@phystech.edu, ORCID: https://orcid.org/0000-0002-1640-1357)

Molodchenkov, Alexey I. — Candidate of Technical Sciences, Federal Research Center "Computer Science and Control" of RAS employee, employee of the Peoples' Friendship University of Russia (e-mail: aim@tesyan.ru, ORCID: https://orcid.org/0000-0003-0039-943X)

Lukin, Anton V. — Federal Research Center "Computer Science and Control" of RAS employee, employee of the Peoples' Friendship University of Russia (e-mail: antonvlukin@gmail.com, ORCID: https://orcid.org/0000-0003-4391-1958)

УДК 004.912
DOI: 10.22363/2658-4670-2023-31-1-64-74
EDN: VNWSXI

Методы извлечения биомедицинских текстов из патентов и научных публикаций (на примере химических соединений)

Н. А. Колпаков[1], **А. И. Молодченков**[2,3], **А. В. Лукин**[2,3]

[1] *Московский физико-технический институт, Институтский переулок, д. 9, Долгопрудный, Московская область, 141701, Россия*
[2] *Федеральный исследовательский центр «Информатика и управление» РАН, ул. Вавилова, д. 44, корп. 2, Москва, 119333, Россия*
[3] *Российский университет дружбы народов ул. Миклухо-Маклая, д. 6, Москва, 117198, Россия*

Аннотация. В данной статье предложен алгоритм для решения задачи извлечения информации из биомедицинских патентов и научных публикаций. Представленный алгоритм основан на методах машинного обучения. Авторами были проведены эксперименты на патентах из базы USPTO. Эксперименты показали, что лучшее качество извлечения продемонстрировала модель, построенная на основе BioBERT.

Ключевые слова: машинное обучение, обработка естественного языка, извлечение именованных сущностей, обработка биомедицинских текстов

Discrete & Continuous Models
& Applied Computational Science
ISSN 2658-7149 (online), 2658-4670 (print)

2023, 31 (1) 75–86
http://journals.rudn.ru/miph

Research article

UDC 538.9
DOI: 10.22363/2658-4670-2023-31-1-75-86
EDN: VTYHBA

Studying the mechanism of electric explosion of metal conductors

Nikolay Yu. Kravchenko, Sergey S. Kovtunov

*Peoples' Friendship University of Russia (RUDN University),
6, Miklukho-Maklaya St., Moscow, 117198, Russian Federation*

(received: March 30, 2023; revised: April 4, 2023; accepted: April 10, 2023)

Abstract. The article gives a description of the history of the development of research of electric explosion of metal conductors, the authors offer a modern view on the physics of the process of electric explosion. The result of such an explosion can be, in particular, the production of nanopowders, which today have found the widest application in industry, agriculture, medicine, and so on.

Key words and phrases: electrical explosion of conductors, explosion physics, exploding wires, nanopowders

1. Introduction

An electric explosion of metal conductors is usually understood to mean the explosive destruction of a conductor with a powerful current impulse passing through it. When a metal conductor is rapidly heated to a temperature above the boiling point, a metal-liquid-vapor phase transition is observed, a dense metal plasma is formed and further formation of small particles, while the explosion products expand and cool. The products of the explosion flying away at high speed quickly cool down, a fine powder is formed. Having created suitable initial conditions for this experiment, particularly, nanopowders are possible to obtain.

Today, nanopowders are widely used as raw materials for the production of ceramic, magnetic, and composite materials, as well as in the production of superconductors, solar cells, lubricant additives, and so on. The use of nanopowders in industry is also very extensive. These are diffusion welding technologies, the creation of protective and anti-friction coatings and the restoration of worn parts of mechanisms. Also, the intensity of the use of nanopowders in agriculture and the environmental protection industry increases annually in the extraction and processing of minerals, in water treatment, in cosmetology and medicine.

This phenomenon is accompanied by current interruption and generation of high-voltage pulses. In addition, the formation of high power shock waves,

the flow of chemical reactions and the formation of bright flashes of light is possible. Due to the fact that an electric explosion is realized under very different conditions (by energy level, by type of metal, by environment, by final result), the resulting effects are also very diverse. All this contributes to a detailed study of the present phenomenon — here, the study of thermophysical properties of metals at high temperatures. An electric explosion of high-temperature plasma can be used as a source in studies of controlled thermonuclear fusion [1, 2], in studies generation of powerful soft X-ray pulses as well as in works on the creation of light sources. In addition, the explosion of wires is used in breakers, fuses in various electrophysical installations.

In connection with the above, more accurate knowledge of the mechanisms of the electrical explosion of metallic conductors is extremely important today. We want to consider the history of the development of this issue and formulate our thoughts on the mechanism of the explosion of conductors to better understand this process.

2. Background

The first theoretical publications on the explosion of conductors date back to the second half of the 18th century. One of the founders of experiments was Michael Faraday in the middle of the 19th century. When a Leyden jar was discharged through a gold wire on the inner walls of the flask, he received a thin metal film. But by the middle of the 20th century the number of publications on the explosion of conductors exceeded 800. It becomes obvious that the electrical explosion of conductors belongs to a poorly studied area of the interaction of metals with electric and magnetic fields. For example, the electrical conductivity of metals has been well studied only in the condensed state and in the ideal state plasma at a temperature of the order of 10^4 K. And such states as the vicinity of the critical point are still the subject of research.

In connection with the foregoing, the study of the mechanism for the realization of an electric explosion of metallic conductors seems to us extremely important.

Nanopowders can be obtained by electrical explosion of a conductor by passing through it a powerful current pulse with a duration of about 1 microsecond and a density from 10^4 to 10^6 A/mm^2. Such impulse heating of the metal can be carried out by discharging a charged capacitor through a thin wire. A wire with a diameter of 0.1 to 1.0 mm is used for this purpose. An electric explosion is accompanied by the generation of shock waves and creates the possibility of rapid heating of metals at a rate of more than $10^8 \div 10^{10}$ K/sec to high temperatures $T > 10^4 K$. At the initial stage of the electric explosion, the heating of the conductor is accompanied by its linear expansion at speed of about 2 m/sec. At the explosion stage, the metal overheats above the melting temperature as a result of the passage of a current pulse, the substance expands at a rate of up to $5 \cdot 10^3$ m/sec. The pressure and temperature of the shock wave front reaches 10^8 Pa [3]. As a result of condensation with the rapid expansion of steam, small particles are formed. Thus, by changing the explosion conditions, it is possible to obtain powders with particle sizes from 100 μm to 50 nm [4]. And by means of an electric

explosion in an inert atmosphere, it is possible to obtain powders of metals and alloys (oxides, nitrides and metal carbides) and fine powders of oxides, nitrides, carbides or mixtures thereof by introducing additional reagents into the reactor (air, a mixture of oxygen and an inert gas, nitrogen, distilled water, paraffin, technical oil). There are known cases of obtaining copper powders by electric explosion in an inert gas at a pressure of 200 Pa with a size of about 20 nm and aluminum powders with an average particle size of about 50 nm. Anderson in his paper [5] photographed line spectra and estimated the temperature of such a plasma $T = 2 \cdot 10^4$ K, which is close to the temperature of stellar atmospheres. There are also known attempts to use this plasma in experiments on the problem of controlled thermonuclear fusion [6].

According to the experimentally obtained data, the powders obtained by the explosion of an electric wire have a large excess energy. So, aluminum powders with an average particle size of 500 to 800 nm have an excess energy of up to 200 kJ/mole and silver powders with an average particle size of about 120 nm have an excess energy of up to 80 kJ/mole, which is several times more than the melting heat of a massive substance. We believe that excess energy cannot be caused only by surface energy. It is believed that the excess energy of fine powders obtained by electric explosion is stored in the form of surface energy, internal defects and charge. The size distribution of the powder particles lies in the range from 10 to 500 nm. The particles of metal powders obtained by electroexplosion are spherical, while the particles of nitride powders have a cut. It is worth mentioning that interest in this described phenomenon is also due to the use of "exploding wires" to obtain a dense high-temperature plasma. The first author to obtain synchronous oscillograms of current and voltage during the discharge of a charged capacitor through a copper conductor was Vrana [7], who discovered that the current in the conductor stops during discharge at current density of more than 10^6 A/sm^3 and after a while resumes, creating a secondary pulse lasting almost until the capacitor is completely discharged (figure 1). The conductor material was found lately to be in the plasma state during the period of the secondary current pulse.

Purposeful studies of the thermophysical properties of metals by the method of electric explosion of conductors were laid down by the work of Soviet scientists. For example, in the works of Lebedev [8] molybdenum, tungsten, nickel and platinum conductors with a diameter of 0.05–0.10 mm were blown up by current pulses with a density of more than 10^6 A/sm^2. Characteristic points t_1, t_2, t_3, t_4 were fixed on the oscillograms of current and voltage on the conductor (figure 1). To clarify the physical nature of the points t_1 and t_2, experiments were carried out with forced switching off of the current at a given time. With the current turned off at time t_1, the conductor remained in a solid state; with turned off at time t_2 is it falls apart into separate drops. This proves that the moments t_1 and t_2 correspond to the beginning and end of the melting of the sample. On the oscillograms for molybdenum and platinum, the energy introduced into the metal is calculated over the time $(t_2 - t_1)$. The metal goes into an anomalous state after the moment t_3, when the stress on the sample rises sharply. The resistance of liquid conductors made of tungsten, platinum, molybdenum and nickel turned out to be weakly dependent on the energy introduced into them in the range from t_2 to t_3.

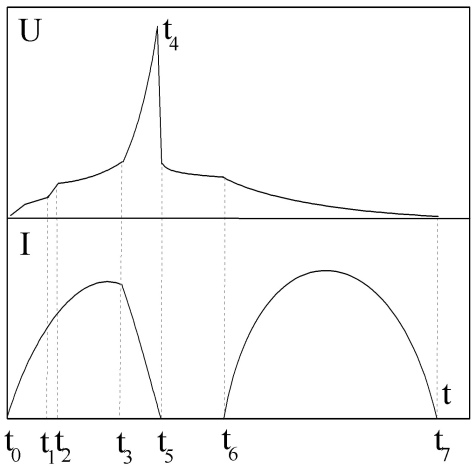

Figure 1. The oscillogram of the current I and voltage U on the conductor in case of an electrical explosion of a copper conductor

During the explosion of refractory metals in air in the range from t_2 to t_3, a violation of the oscillograms is observed. This is due to the development of a gas discharge in air along the surface of the sample and leads to heating of the metals. There are various mechanisms of destruction of the conductor: breaks, disintegration into drops, destruction with the formation constrictions, uniform expansion of the liquid conductor in the initial stage of its explosion and subsequent formation of a transverse layered structure were found to be possible by using the method of obtaining single photographs of a conductor being heated by a current pulse.

In Kvartskhava's experiments [9] the energy being introduced into a copper conductor during the first current pulse was measured. This energy in the case of microsecond pulses appeared to increase with increasing current density and can exceed the heat of metal evaporation. The resistance of the conductor after t_3 for the energy given decreases with the current density increasing. In the research of Kulgavchuk and Novoskoltseva X-ray images of an exploding copper synchronized with current and voltage oscillograms conductor were obtained.

The conductor was found to expand after the point t_3 in the case of microsecond current pulses remaining intact. Its stratification into transverse layers begins in the vicinity of the point t_4, after the density of copper in a solid conductor becomes equal to $\rho = 4.1$ g/sm^3, i.e. in similar works by foreign authors [10–13] the main attention is paid study of the process of electric explosion and shock waves. The shock wave begins to develop after the moment t_3 (figure 1), so the point t_3 can be considered the starting point of the electrical explosion. Peregud [14] studied the development process of magnetohydrodynamic and electrothermal instabilities of the shape of a metal conductor being heated by a current pulse. They also detected luminescent radiation of metals (figure 1) emitted by a liquid conductor being destroyed at the end of the first current pulse.

3. Mechanism of electrical explosion of conductors

To heat the conductor to the starting point of the electric explosion of the conductor (point t_3 on figure 1) without destroying the subsequent explosive boiling of the metal it is necessary to satisfy three basic conditions.

1. The heating of the conductor must be continuous, which requires the aperiodicity of the discharge of the capacitor. This means that the average resistance of the conductor during the heating process must satisfy the condition:

$$R \geqslant 2\left(\frac{L}{C}\right)^{\frac{1}{2}}, \tag{1}$$

where L, C are inductance and capacitance of the capacitor. If we assume that the resistance R of a liquid conductor at melting temperature $R_2 = \rho_2 l/s$, then it follows from condition (1) that the ratio of the conductor length l to its cross section s :

$$\frac{l}{s} \geqslant \frac{2}{\rho_2}\left(\frac{L}{C}\right)^{\frac{1}{2}}, \tag{2}$$

where ρ_2 is the resistivity of the liquid metal at the melting point.

2. The second condition is such that when heated, the conductor should not be destroyed under the influence of MHD perturbations of its shape and evaporation from its surface. To do this, it must be heated, observing the conditions:

$$\varepsilon \geqslant \varepsilon_m = \frac{i_m^2 \rho}{DC_p} = \frac{\mu_o C_p \left(T_3 - T_2\right)^2}{4\rho}, \tag{3}$$

where ρ, D, C_p are the average values of the resistivity, density and specific heat capacity of a liquid metal in the temperature range from T_2 to T_3. This velocity is related to the time t_3 of heating the conductor to the point T_3, for which the condition must be met:

$$t_3 \leqslant t_{3m} = \frac{T_3 - T_0}{\varepsilon_m}, \tag{4}$$

where ε_m is the greater of the values ε_m, $T_0 = 300$ K is the initial temperature of the conductor.

Let the point T_3 be reached at time t_m of maximum current in the circuit. If the discharge occurs in a way close to critical, then $t_3 = t_m = 2L/R$. Substituting in this expression instead of R the resistance of the conductor at the melting point, from (3) the second condition is obtained:

$$\frac{l}{s} = \frac{2}{\rho_2} \cdot \frac{L\varepsilon_m}{T_3 - T_0}. \tag{5}$$

3. The third condition is the sufficiency of energy charged capacitor. If this energy is sufficient not only to heat the conductor to a temperature of T_3, but also to overheat it to a temperature of T_4, then its intense

explosive boiling will occur. But will arise in constrictions in which the electrothermal instability develops after the point T_3. Assuming that at the point T_4 the specific energy of the metal is ϖ_4, then

$$0,5 \cdot C \cdot U_0^2 \geqslant \varpi_4 m, \tag{6}$$

where U_o is the initial voltage on the capacitor; m is the mass of the conductor.

Thus, the conditions (2), (5), (6) are well satisfied for conductors whose l/s ratio is large enough, provided that the discharge circuits have a low inductance and a high initial voltage across capacitor. Such conditions are easier to fulfill for metals with bigger resistivity.

The table 1 presents the results of calculating the minimum current density and the heating rate of the conductor according to the formula (3), as well as the maximum heating time t_{23m} of a liquid conductor from the point T_2 to the point T_3 for different metals, when calculating $T_3 = 0.80 \cdot T_c$ is accepted.

Table 1

Boundary conditions for the parameters of pulsed heating of metals taking into account evaporation from the surface of the conductors

	W	Mo	Pt	Cu	Pb	Cs	Hg
$i_m \cdot 10^{-6}$,A/sm^2	2.30	3.12	3.39	7.47	1.68	1.63	3.74
$\varepsilon_m \cdot 10^{-8}$, K/sec	4.10	5.22	3.53	5.93	2.80	8.67	10.2
t_{23m}, μsec	20.7	11.9	12.5	5.52	11.0	1.53	1.09

4. Out-spinodal electrical explosion and spinodal decay unstable liquid metal phase

In figure 2 a diagram of the states of titanium in the liquid-vapor phase transition region is presented. On it, C is the critical point calculated by the equation (1); for this point, according to [15], $T_c = 9040$ K, $p_c = 156$ MPa, $V_s = 71.1$ sm^3/mole. The binodal bC is determined by extrapolating the temperature dependence of the saturated vapor pressure from the normal boiling point T_2, p_2 to the critical point T_c, p_c; it is calculated by the formula

$$\ln \frac{p}{p_2} = B - \frac{A}{T}, \tag{7}$$

in which the constants A and B are equal:

$$A = \frac{\ln (p_c/p_2)}{1/T_2 - 1/T_c}, \quad B = \frac{A}{T_2}.$$

For titanium $A = 4.34 \cdot 10^4$ K, $B = 12.14$.

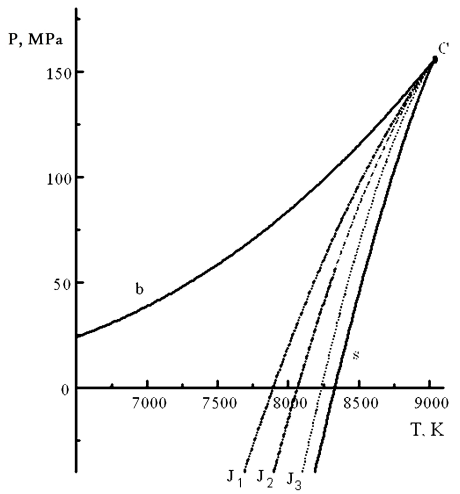

Figure 2. The diagram of the states of titanium in the liquid-vapor phase transition area

Spinodal sC of the liquid phase in figure 2 is crosses the temperature axis at the point $T_{s,p=0} = 8330$ K, then it enters the field of negative pressures [16].

The area of the metastable liquid (superheated or stretched liquid) lies between the binodal and the spinodal. In the area of positive pressures, the spinodal determines the boundary of the thermodynamic stability of the liquid. In the area of negative pressures [17], the spinodal determines the tensile strength of the liquid.

The appearance of an overheated liquid metal near the spinodal was experimentally proved when metal conductors are heated by a microsecond current pulse. In this case, the metal evaporates both through the surface of the conductor and through the surface of the vapor nuclei that appear on the finished centers. However, such evaporation is insignificant and therefore the conductor remains in the liquid state until the spinodal is reached. The calculation shows [18] that the minimum heating rate of various metals to near-spinodal states is equal to $(3 \div 10) \cdot 10^8$ K/sec, which corresponds to the density of the heating current $(2 \div 8) \cdot 10^6$ A/sm^3 and the heating time of the liquid $1 \div 10$ μsec. With pulsed heating of the conductor at atmospheric pressure the heating line is located below the binodal line bc in figure 2, since the reactive pressure of the scattering vapor acts on the surface of the conductor $p = 0.55 p_b$ [18]. Overheating of a thermodynamically stable liquid above the spinodal point is impossible, since when approaching this point, an explosive boiling mechanism is activated, caused by a high frequency of homogeneous nucleation of vapor nuclei.

Figure 2 shows lines for three different frequencies of homogeneous nucleation of vapor nuclei in superheated liquid titanium [19]:

$$J_1 = 1 \text{ sm}^{-3} \cdot \text{sec}^{-1}, \quad J_2 = 10^{20} \text{ sm}^{-3} \cdot \text{sec}^{-1}, \quad J_3 = 10^{28} \text{ sm}^{-3} \cdot \text{sec}^{-1}.$$

The values required for calculations ρ_L, ρ_v, σ are determined by empirical formulas for the thermodynamic properties of liquid metals near the critical point [20]. The frequency of homogeneous nucleation is shown by calculations

to increase by 28 orders of magnitude when approaching the spinodal along the heating line at $p = 0.55p_b$ in the temperature range $\Delta T = 230$ K $= 0.025T_c$. This ensures explosive effervescence of superheated liquid metal without reaching a spinodal phase explosion. Thus, explosive boiling up is the main factor determining the electrical explosion of conductors when heated by a microsecond current pulse.

Thus, if during pulsed heating of a liquid, its heating time satisfies the condition $\Delta t_s < \theta$, then it becomes possible to enter the region of an unstable phase (behind the spinodal), while the thermodynamic stability coefficients are negative. The elasticity and heat capacity of such a phase will also be negative. The surface tension of the unstable liquid phase is zero [21], which limits the formation of vapor nuclei. These features determine the nature of the decay of the unstable phase, known as spinodal decay [22]. With the subsequent development of the described process, a continuous formation of an heterogeneous structure occurs, consisting of small contracting and expanding regions without phase boundaries between them. It is believed that the speed of a spinodal decay is determined by heat exchange between the described areas. Since the heat capacity of the unstable phase is negative, its thermal conductivity is $a = \lambda/\rho \cdot C_p$ also negative (λ is the thermal conductivity coefficient). The temperature differences arising in the locally unstable phase increase exponentially until the system leaves the zone of instability. The process of spinodal decomposition is replaced by the formation and growth of nuclei of a new phase.

The possibility of realizing unstable states of liquid metals is evidenced by experiments with an "anomalous" electric explosion of conductors at a heating rate more than $5 \cdot 10^{10}$ K/sec and heating time to the peak voltage point $t_4 < 0.1$ μsec.

Under these conditions, the heating time of the liquid metal Δt_S in the vicinity of the spinodal will be less than the time development of the process of homogeneous nucleation of vapor nuclei, which determines the possibility of reaching spinodal [23]. The energy was shown to be W_4 introduced into the metal with this heating mode by the time t_4 is several times higher than the heat of its evaporation Λ_{b0} at the normal boiling point. For example, with an abnormal explosion of gold conductors at a current density of $i = 3.3 \cdot 10^8$ A/sm^2 the ratio of $W_4/\Lambda_{b0} = 10$. At the same time the volume of metal is shown by method of high-speed photography to exceed the initial volume by no more than 4 times in the vicinity of the point t_4. A large energy density was shown by these experiments that are carried out in an unstable liquid phase for a short time (several nanoseconds). This ensures high attenuation intensity and high local temperatures at individual points.

The existence of such local "hot spots" during the electrical explosion of conductors is confirmed by the release of multiply charged ions from the explosion zone; for example, charge states of atoms were found: Cu^{+27}, Ag^{+37}, Au^{+51}. In [24] during an out-spinodal explosion of aluminum, the release of X-ray quantum with the energy 1–20 keV was detected. The output of short-wave X-ray quantum with a wavelength $0.15 \div 0.28$ nm during an electric explosion of titanium and iron was also detected in the work [25]. The rate of expansion of a substance during an out-spinodal electric explosion can be estimated using the formula [26]:

$$v = 2 \cdot \left(\frac{\gamma \varpi_4}{\gamma - 1} \right)^{1/2}, \tag{8}$$

where γ is adiabatic index; ϖ_4 is excess specific energy introduced into the metal during its overheating from the normal boiling point to the point t_4 (figure 1). With $\gamma = 1.2$ [26] and the values of ϖ_4 obtained in experiments with an "anomalous" electric explosion of conductors made of copper, silver, gold and tin [18], $v = (15 - 20)$ km/sec. The so-called "anomalous" electric explosion is of great interest. At a sufficiently high heating rate, energy, being several times higher than the heat metals sublimation, enters the metal. In the process of such an "anomalous" electrical explosion, an emission of X-rays and multiply charged ions was detected. X-ray radiation ejected from an exploding conductor during its expansion in 10 nsec was showed by the photographs and observed with a high-speed video camera. This is explained as follows. In the process of spinodal decomposition of the liquid metal phase, regions with local elevation points appear. This leads to thermal excitation of atoms and electronic transitions generating X-ray quantum.

5. Conclusions

To summarize, we analyzed the literature on the history of the formation and development of studies of the explosion of metal conductors and offered an understanding of the mechanism of this process.

References

[1] M. M. Martynyuk and N. Y. Kravchenko, "Nuclear fusion reaction in metaphase substance in the process of electrical explosion [Reaktsiya yadernogo sinteza v mezofaznom veshchestve v protsesse elektricheskogo vzryva]," *Applied Physics*, no. 1, p. 79, 2003, in Russian.

[2] M. M. Martynyuk and N. Y. Kravchenko, "Impact-cluster nuclear fusion. Conditions of excitation of the process [Udarno-klasternyy yadernyy sintez. Usloviya vozbuzhdeniye protsessa]," *Bulletin of Peoples' Friendship University of Russia. Series: Mathematics, Informatics, Physics*, no. 1, pp. 118–128, 2005, in Russian.

[3] N. Y. Kravchenko, "The numerical solution of the Rayleigh–Plisset equation for spark cavitation and calculation of the maximum temperature and pressure in a cavity," *Journal of Mechanics of Continua and Mathematical Sciences*, no. Special Issue-1, pp. 465–473, Mar. 2019. DOI: 10.26782/jmcms.2019.03.00046.

[4] N. Y. Kravchenko and M. M. Martynyuk, "Dynamics of homogeneous cavitation bubbles in water under large amplitude pressure oscillations [Dinamika zarodyshey gomogennoy kavitatsii v vode pod deystviyem davleniya bol'shoy amplitudy]," *Bulletin of Peoples' Friendship University of Russia. Series: Mathematics, Informatics, Physics*, vol. 8, pp. 118–121, 2000, in Russian.

[5] J. A. Anderson and S. Smith, "General charcteristics of electrically exploded wires," *The Astrophysical Journal*, vol. 64, no. 5, pp. 295–314, 1926.

[6] L. A. Artsimovich, *Controlled thermonuclear reactions [Upravlyayemyye termoyadernyye reaktsii]. Ed. 2nd.* M.: Fizmatgiz, 1963, in Russian.

[7] J. Wrana, "Vorghnge beim Schmeilzen and Verdampfen von Drghten mit sehr hochen Stromdichten," German, *Archiv für Electrotechnik*, vol. 33, no. 10, pp. 656–672, 1939.

[8] S. V. Lebedev, "Explosion of a metal by an electric current," *Journal of Experimental and Theoretical Physics*, vol. 32, no. 2, p. 199, 1957.

[9] I. F. Kvartskhava, A. A. Plyutto, A. A. Chernov, and V. V. Bondarenko, "Electrical explosion of metal wires," *Journal of Experimental and Theoretical Physics*, no. 1, p. 40, 1956.

[10] A. A. Rukhadze, Ed., *Exploding wires [Vzryvayushchiyesya provolochki].* 1963, in Russian.

[11] A. A. Rukhadze and I. S. Spiegel, Eds., *Electric explosion of conductors [Elektricheskiy vzryv provodnikov].* M.: Mir, 1965, in Russian.

[12] W. G. Chace and H. K. Moore, Eds., *Exploding wires.* N.Y.: Plenum Press, 1964, vol. 3.

[13] W. G. Chace and H. K. Moore, Eds., *Exploding wires.* N.Y.: Plenum Press, 1968, vol. 4.

[14] K. B. Abramova, N. A. Zlatin, and B. P. Peregud, "Magnetohydrodynamic instability of liquid and solid conductors. Destruction of conductors by an electric current," *Journal of Experimental and Theoretical Physics*, vol. 6, no. 12, p. 2007, 1975.

[15] P. A. Tamanga, M. M. Martynyuk, and N. Y. Kravchenko, "Spinodal of liquid phase on basis of generalized Berthelo's equation [Spinodal' zhidkoy fazy po obobshchennomu uravneniyu Bertelo]," *Bulletin of Peoples' Friendship University of Russia. Series: Mathematics, Informatics, Physics*, no. 9, pp. 56–58, 2001, in Russian.

[16] M. M. Martynyuk and N. Y. Kravchenko, "Limit of thermodynamic stability of a liquid phase in the field of negative pressure [Granitsa termodinamicheskoy ustoychivosti zhidkoy fazy v oblasti otritsatel'nykh davleniy]," *Journal of Physical Chemistry*, vol. 72, no. 6, pp. 998–1001, 1998, in Russian.

[17] M. M. Martynyuk and N. Y. Kravchenko, "Boundary of liquid phase thermodynamic stability at negative pressure," *Russian Journal of Physical Chemistry*, vol. 72, no. 6, pp. 885–887, 1998.

[18] M. M. Martynyuk, *Phase transitions during pulsed heating [Fazovyye perekhody pri impul'snom nagreve].* Moscow: RUDN, 1999, in Russian.

[19] M. M. Martynyuk, P. A. Tamanga, and N. Y. Kravchenko, "The titanium phase diagram at the phase transition region liquid-vapor [Fazovaya diagramma titana v oblasti fazovogo perekhoda zhidkost'-par]," *Bulletin of Peoples' Friendship University of Russia. Series: Mathematics, Informatics, Physics*, no. 10, pp. 121–125, 2002, in Russian.

[20] M. M. Martynyuk, *Parameters of the critical point of metals [Parametry kriticheskoy tochki metallov]*. M.: RUDN, 1989, in Russian.

[21] M. M. Martynyuk and N. Y. Kravchenko, "Conditions for excitation of spinodal decomposition of an unstable liquid phase in the process of pulsed heating of metals," *Bulletin of the PFUR. Series Mathematics, Informatics, Physics*, no. 1, pp. 92–99, 2008, in Russian.

[22] V. P. Skripov and A. V. Skripov, "Spinodal decomposition (phase transitions via unstable states)," *Sov. Phys. Usp.*, vol. 22, pp. 389–410, 1979. DOI: 10.1070/PU1979v022n06ABEH005571.

[23] N. Y. Kravchenko and M. M. Martynyuk, "Dynamics of homogeneous cavitation nuclei in water under high-amplitude sinusoidal pressure fluctuations [Dinamika odnorodnykh yader kavitatsii v vode pri vysokoamplitudnykh sinusoidal'nykh pul'satsiyakh davleniya]," *Bulletin of PFUR. Series Physics*, no. 8 (1), pp. 118–121, 2000, in Russian.

[24] I. M. Vitkovitsky, "X-ray emission from exploding wires," *Physics of Fluids*, vol. 7, no. 4, p. 612, 1964. DOI: 10.1063/1.1711249.

[25] P. Burkhalter, J. Davis, J. Rauch, and W. Clark, "X-ray line spectra from exploded-wire arrays," *Applied Physics*, vol. 50, no. 2, p. 705, 1979. DOI: 10.1063/1.326034.

[26] Y. B. Zeldovich and Y. P. Raizer, *Physics of shock waves and high-temperature hydrodynamic phenomena [Fizika udarnykh voln i vysokotemperaturnykh gidrodinamicheskikh yavleniy]*. M.: Nauka, 1966, in Russian.

For citation:

N. Y. Kravchenko, S. S. Kovtunov, Studying the mechanism of electric explosion of metal conductors, Discrete and Continuous Models and Applied Computational Science 31 (1) (2023) 75–86. DOI: 10.22363/2658-4670-2023-31-1-75-86.

Information about the authors:

Kravchenko, Nikolay Yu. — Candidate of Sciences in Physics and Mathematics, Deputy Director of Institute of Physical Research and Technology, RUDN University (e-mail: kravchenko-nyu@rudn.ru, phone: +7(495)9550839, ORCID: https://orcid.org/0000-0003-3397-1746, ResearcherID: E-6162-2018, Scopus Author ID: 14633789300)

Kovtunov, Sergey S. — Student of Department of Mechanics and Management Processes, Engineering Academy, RUDN University (e-mail: 1032216305@rudn.ru, phone: +7(929)0232607, ORCID: https://orcid.org/0009-0004-4401-6167)

УДК 538.9
DOI: 10.22363/2658-4670-2023-31-1-75-86
EDN: VTYHBA

Изучение механизма электрического взрыва металлических проводников

Н. Ю. Кравченко, С. С. Ковтунов

Российский университет дружбы народов,
ул. Миклухо-Маклая, д. 6, Москва, 117198, Россия

Аннотация. В статье представлено описание истории развития исследований электровзрыва металлических проводников, предложен современный взгляд на физику процесса электровзрыва. Результатом такого взрыва может стать, в частности, производство нанопорошков, которые сегодня нашли самое широкое применение в промышленности, сельском хозяйстве, медицине и так далее.

Ключевые слова: электрический взрыв проводников, физика взрыва, взрывающиеся провода, нанопорошки