considered that a correct description of the propagation of Lamb wave packets is possible only for the case of orthotropic composite materials. The possibility of taking into account only low monoclinic symmetry for the excitation mode and propagation of wave packets in a transversely isotropic laminate along a non-principal direction should be attributed to the advantages of this method.

References:

1. Ebrahiminejad, A., Mardanshahi, A., & Kazemirad, S. (2022). Nondestructive evaluation of coated structures using Lamb wave propagation. Applied Acoustics, issue 185, p. 108378
DOI: 10.1016/j.apacoust.2021.108378

2. Su, Z., Ye, L., Su, Z., & Ye, L. (2009). Fundamentals and analysis of lamb waves. Identification of Damage Using Lamb Waves: From Fundamentals to Applications, pp. 15-58
DOI: 10.1007/978-1-84882-784-4_2

3. Yang, C., Ye, L., Su, Z., & Bannister, M. (2006). Some aspects of numerical simulation for Lamb wave propagation in composite laminates. Composite structures, issue 75, vol. 1-4, pp. 267-275
DOI: 10.1016/j.compstruct.2006.04.034

4. Peddeti, K., & Santhanam, S. (2018). Dispersion curves for Lamb wave propagation in prestressed plates using a semi-analytical finite element analysis. The Journal of the Acoustical Society of America, issue143, vol. 2, 829-840
DOI: 10.1121/1.5023335

Article sent: 20.05.2025

https://www.sworldjournal.com/index.php/swj/article/view/swj31-02-081
DOI: 10.30888/2663-5712.2025-31-02-081

UDC 004.42:519.834:519.72

AUTOMATION OF THE PROCESS OF STUDYING CONFLICT SITUATIONS AND SOLVING GAME THEORY PROBLEMS WITH LINEAR PROGRAMMING METHODS

Kondratieva N.O.
c.ph.-m.s., as. prof.
ORCID: 0000-0002-6994-2536
Leontieva V.V.
c.ph.-m.s., as. prof.
ORCID: 0000-0002-9863-9712
Pazinich K.A.
4-year student
Zhelobetskiy A.P.
4-year student
Usatenko G.G.
postgraduate student
Gorbachov O.A.
postgraduate student
Zaporizhzhia National University,
Universytetska Str., 66, Zaporizhzhia, Ukraine, 69063

Abstract. *The research focuses on the automation of analyzing conflict situations through the effective solution of game theory problems. The article presents and substantiates a methodology for reducing mixed-strategy matrix games to linear programming problems. We developed and experimentally verified software that automates the formulation and solution of corresponding optimization tasks. This software enables rapid and highly accurate calculation of optimal strategies and game value, demonstrating scalability for handling large-dimensional games. The work's main contribution lies in creating a universal tool that provides mathematically sound recommendations for decision-making, minimizing risks and maximizing payoffs across various applied fields.*

Key words: *conflict situation, matrix game, mixed strategies, game theory, linear programming, automation, software.*

Introduction.

In a dynamic and interdependent world where system complexity constantly increases, the ability to make optimal and strategically sound decisions is a key factor for success. Decision-making in situations of such strategic interaction is one of the central and most complex problems across many fields of science and engineering. These decisions often demand a deep understanding of the interactions between numerous participants, as well as the prediction of their actions and potential reactions. In these conditions, it is possible to arise conflict situations where the outcome depends on a set of interrelated choices, the choice of each participant directly affects the gains of others, and the choice of the optimal strategy is made in the face of a conflict of interest. Such conflict situations involving multiple participants, each of whom seeks to maximize their own objective function, are an integral part of social, economic and political processes, arise regularly and require grounded methods of analysis and resolution. Analysis and understanding of conflict situations is an important task for making informed decisions and developing management strategies. Effective identification of the best strategies and forecasting of stable states in these scenarios are critical, as without adequate analytical tools, this leads to significant uncertainty, potentially ineffective decisions, and significant losses of resources. In such circumstances, the outlined task is a key task, but at the same time a computationally and analytically complex task that requires precise analytical tools and an automated approach to its solution.

To structure and evaluate complex decision-making problems, system analysis and decision-making theory provide general methodological foundations. However, when optimal decisions depend on the interaction of several rational participants in a conflict situation, game theory gains particular relevance. Game theory is a mathematical framework for modeling decision-making processes under conditions of conflict and uncertainty, and therefore serves as a tool for formalizing and analyzing these strategic interactions [1, 2]. Of particular importance in this context are matrix games (normal-form games) – a class of games with a finite number of players and discrete sets of strategies, which can be represented as a payoff matrix [1-5]. Yet, despite their structural clarity, the practical solution of matrix games, especially for large-sized matrices, poses a significant computational challenge. Finding optimal pure or mixed strategies and determining Nash equilibrium often involves solving complex systems of linear inequalities and employing iterative methods. This makes manual analysis impractical on a real-world scale and significantly limits their applicability in practical tasks [6]. In such cases, the mathematical complexity of game theory problems, especially when mixed strategies are used, necessitates the development of efficient algorithms and their computer implementation. The method of reducing game problems to linear programming problems deserves particular attention here, as it allows for the use of well-developed mathematical tools and existing software for their solution [2, 4, 7]. However, automating this process requires a systematic approach that combines mathematical modeling, optimization methods, and computer technologies.

The purpose of research is to develop and substantiate an approach to automate the process of analyzing conflict situations by reducing game theory problems with mixed strategies to linear programming problems, and to create corresponding software.

The object of the study is the decision-making process under conditions of conflict and uncertainty, modeled using the mathematical apparatus of game theory.

The subject of the study is methods and algorithms for solving matrix games with mixed strategies using linear programming techniques.

Theoretical Foundations for Modeling Conflict Situations and Solving Matrix Games.

We will now outline the core theoretical and methodological aspects of applying a systematic game theory approach to analyze strategic interactions among decision-makers in situations of conflicting interests. In today's world, where interactions between various participants in conflict situations often take on an antagonistic character, game theory stands as an indispensable tool for understanding and predicting such behavior. It allows for the formalization of a conflict situation, identifying the players, their possible strategies, and evaluating the consequences of each choice.

Among the numerous models offered by this theory, matrix games hold a special place. Currently, they represent one of the most common classes of game-theoretic mathematical models and are presented as two-person zero-sum antagonistic games, where the gain of one player equals the loss of the other, and the conflict situation itself is represented as a payoff matrix (game matrix) $A = (a_{ij})_{m \times n}$ [4].

However, not all games have a straightforward solution. When a game lacks a saddle point in pure strategies, it becomes necessary to use mixed strategies. This means employing a probability distribution over the set of pure strategies. In other words, players don't choose one specific action but rather use probabilities to select each of their available options. This approach ensures the solvability of many complex conflict situations. According to von Neumann's fundamental theorem, every matrix game has a solution in mixed strategies, corresponding to a specific equilibrium situation [5]. Such a solution indicates a stable state where no player can improve their outcome by unilaterally deviating from their chosen mixed strategy.

Mathematically, the mixed strategy of the first player is represented as a probability vector $p=(p_1, p_2, ..., p_m)$, and the second player's strategy as a vector $q=(q_1, q_2, ..., q_n)$, where $p_i, q_j \geq 0$, $\sum_{i=1}^{m} p_i = 1$, $\sum_{i=1}^{n} q_j = 1$. Under these conditions, the expected payoff for the first player is determined as $E(p,q)=\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} p_i q_j = 1$, where a_{ij} are the elements of the payoff matrix $A=(a_{ij})_{m\times n}$.

To find optimal mixed strategies, we need to solve a minimax problem. This can be computationally complex, especially for games with many strategies. This is where linear programming methods prove effective, offering a powerful tool for solving these problems. We can reduce matrix games with mixed strategies to linear programming problems, allowing us to use the well-developed simplex method to find optimal strategies. This offers significant computational advantages [3, 5, 6, 8].

The reduction process is based on the fact that finding players' optimal mixed strategies can be represented as a pair of dual linear programming problems [4, 9, 10]:

– for the first player:

$$f(x)=\sum_{i=1}^{m} x_i = \frac{1}{V} \to \min, \ \sum_{i=1}^{m} a_{ij} x_i \geq 1, \quad j=\overline{1,n}, \ x_i \geq 0, \ i=\overline{1,m}.$$

– for the second player:

$$f(y)=\sum_{j=1}^{n} y_j = \frac{1}{V} \to \max, \ \sum_{j=1}^{n} a_{ij} y_j \leq 1, \quad i=\overline{1,m}, \ y_j \geq 0, \quad j=\overline{1,n}.$$

The connection between the solutions of these problems and the players' optimal strategies is established as follows: if $x^* = (x_1^*, x_2^*, ..., x_m^*)$ is the optimal solution to the primal problem with an optimal value of $f^*(x)$, then the first player's optimal strategy is determined as $p^* = \left(\frac{x_1^*}{f^*(x)}, \frac{x_2^*}{f^*(x)}, ..., \frac{x_m^*}{f^*(x)} \right)$, and the value of the game is $V = \frac{1}{f^*(x)}$. Similarly for the second player [7]: if $y^* = (y_1^*, y_2^*, ..., y_n^*)$ is the optimal solution to the dual problem with an optimal value of $f^*(y)$, then the second player's optimal strategy is determined as $q^* = \left(\frac{y_1^*}{f^*(y)}, \frac{y_2^*}{f^*(y)}, ..., \frac{y_n^*}{f^*(y)} \right)$, and the value of the game is $V = \frac{1}{f^*(y)}$. Thus, by solving a pair of defined mutually dual symmetric problems, we find $p_i = V \cdot x_i^* \quad \left(i = \overline{1,m}\right)$, $q_j = V \cdot y_j^* \quad \left(j = \overline{1,n}\right)$ and $V = \frac{1}{\sum_{i=1}^{m} x_i^*} = \frac{1}{\sum_{j=1}^{n} y_j^*}$, which allow us to determine the game's solution $(p^*, q^*, V) = \left((p_1, ..., p_m), (q_1, ..., q_m), V\right)$. Such approach offers significant advantages, as it enables the use of advanced methods and software tools for solving large-scale linear programming problems [8].

Computer Modeling and Software Automation.

Effective automation and computer modeling of the process of solving game theory problems using linear programming methods requires the development of specialized software. In this work, we've developed a software product that implements the full cycle of solving the investigated matrix game: from inputting data to interpreting the obtained results.

Python was chosen as the programming language for implementing the matrix game solution task. This language is an optimal choice for solving complex mathematical problems due to its powerful mathematical apparatus and wide range of specialized libraries. Python offers unique capabilities for scientific computing, particularly through the NumPy and SciPy libraries, which enable efficient execution of complex linear optimizations and matrix manipulations. The built-in linprog function from the SciPy library provides a precise algorithm for solving linear programming problems, which is critically important for finding optimal strategies in game theory. Python's advantages when solving such problems also include high performance, simple syntax, and the ability to represent mathematical algorithms in code as closely as possible.

PyCharm from JetBrains was chosen as the development environment. This decision was driven by its powerful debugging tools, intelligent code auto-completion, advanced refactoring capabilities, and seamless integration with version control systems. PyCharm provides a professional toolkit that significantly boosts developer productivity and the quality of the final software product.

The developed matrix game-solving program features a modular structure, built around key mathematical modeling functions. The architecture of the software includes the following modules: a module for forming the payoff matrix and checking for a saddle point in pure strategies, a module for transforming the payoff matrix (if necessary, when the matrix contain negative values), a module for forming mutually dual linear programming problems for both players, a module for solving linear programming problems (using the simplex algorithm), a module for interpreting the results, which includes verifying result consistency and correcting the game's value (if the matrix was transformed).

The program's input data is the game's payoff matrix, provided in a text file where each row contains numerical values separated by spaces. The program's output data includes the optimal strategies for the first (p^*) and second (q^*) players, the value V of the game, and detailed information about the intermediate solution steps.

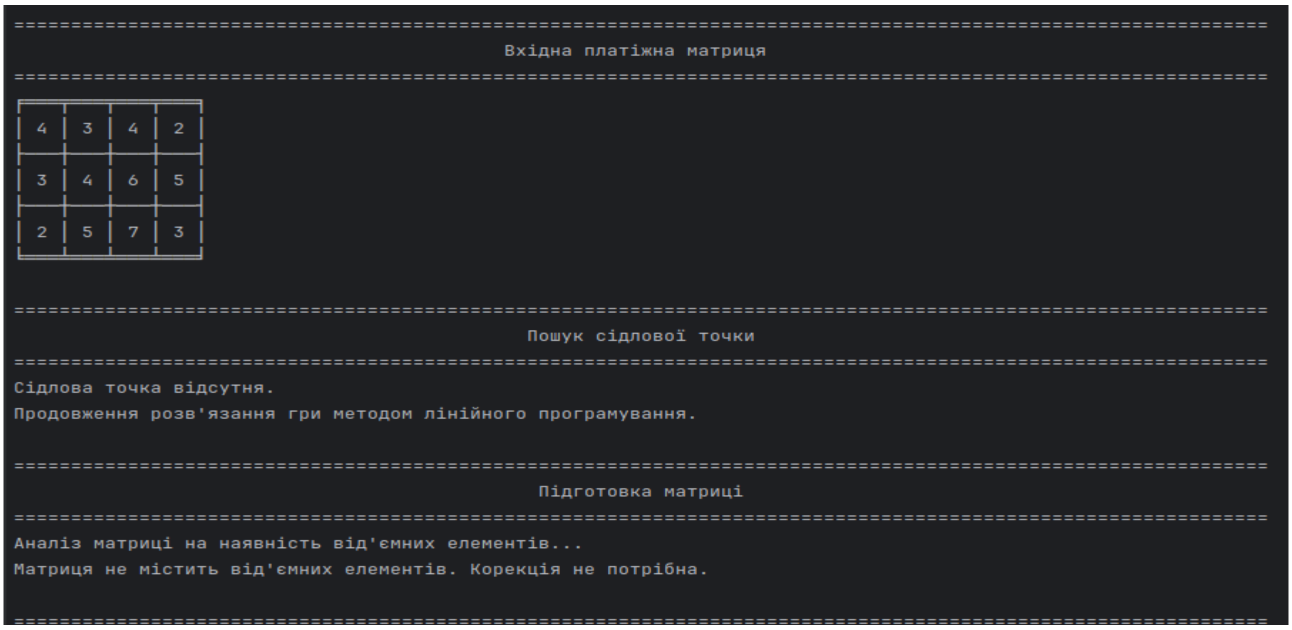
The program's interface is implemented as a console application with detailed text output of results. Users can immediately see all intermediate stages of the matrix game solution, making the process maximally transparent and understandable.

To use the program, you need to install the necessary libraries: NumPy, SciPy, and Tabulate. Then, prepare a text file with the game matrix, where each row represents a matrix row, and numbers are separated by spaces. Finally, run the script via the command line, passing the filename as an argument (e.g., python matrix_game_solver.py game_matrix.txt). The program will automatically read the matrix from the file, perform all necessary calculations, and display the detailed results of the matrix game solution.

Experimental Verification and Software Approval.

We conducted a series of computational experiments to verify the presented algorithmic approach and test the developed software. These experiments used both classic test cases from literature and real-world practical problems from various domains.

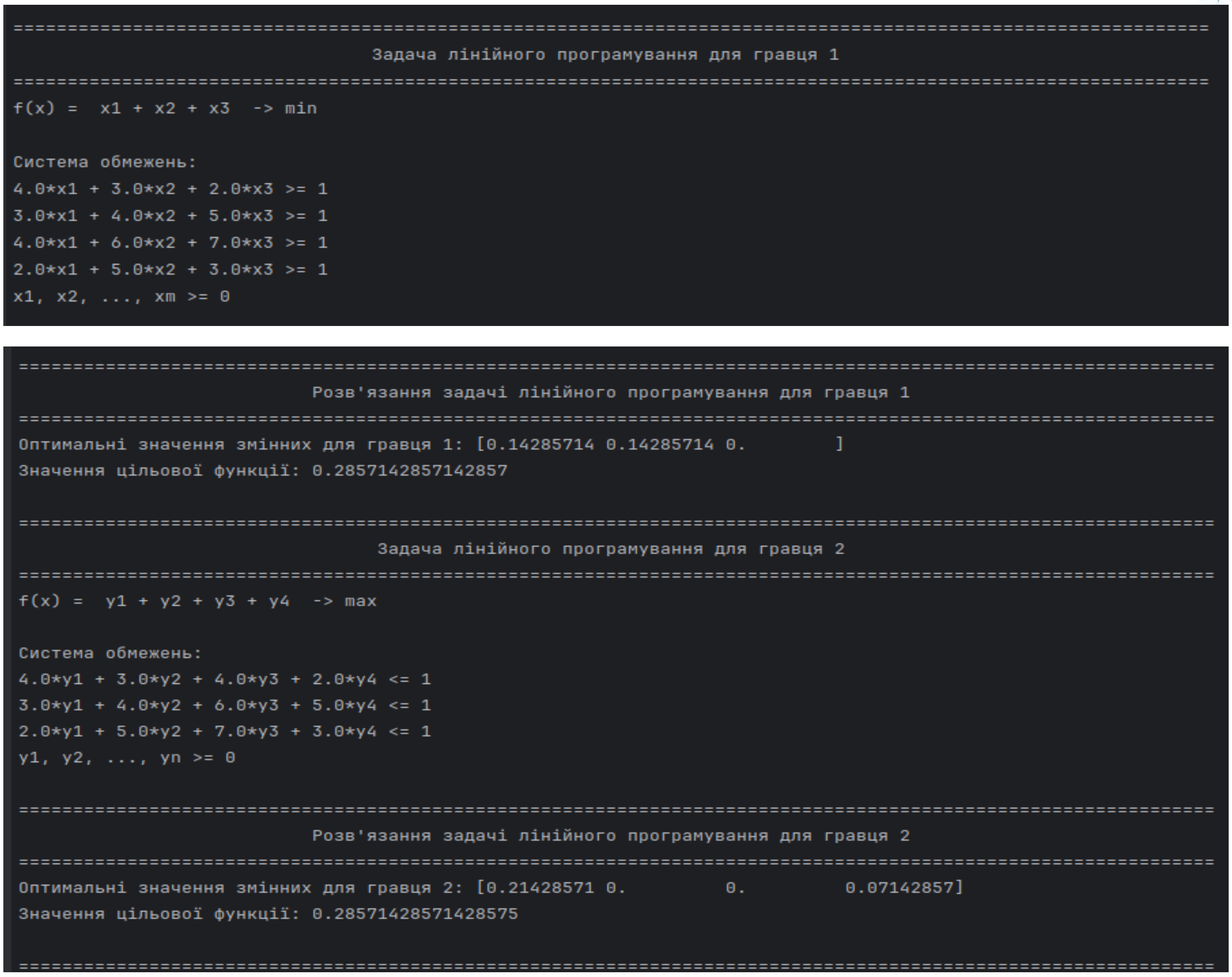
Let's examine one such experiment using a game with a 3×4 payoff matrix. We'll show a step-by-step solution to this problem using our developed software. Figures 1 and 2 display screenshots of the program in action. These screenshots show: reading the input matrix from a user-provided text file, checking the matrix for a saddle point, determining if the payoff matrix needs transformation (based on an analysis for negative values), constructing and solving the primal and dual linear programming problems for both players in the matrix game (the participants in the conflict situation being studied).



```
==========================================================================================
                              Вхідна платіжна матриця
==========================================================================================
┌───┬───┬───┬───┐
│ 4 │ 3 │ 4 │ 2 │
├───┼───┼───┼───┤
│ 3 │ 4 │ 6 │ 5 │
├───┼───┼───┼───┤
│ 2 │ 5 │ 7 │ 3 │
└───┴───┴───┴───┘

==========================================================================================
                              Пошук сідлової точки
==========================================================================================
Сідлова точка відсутня.
Продовження розв'язання гри методом лінійного програмування.

==========================================================================================
                               Підготовка матриці
==========================================================================================
Аналіз матриці на наявність від'ємних елементів...
Матриця не містить від'ємних елементів. Корекція не потрібна.

==========================================================================================
```

Figure 1 – Screenshot of the program's operation showing input and analysis of the game's payoff matrix

Authoring

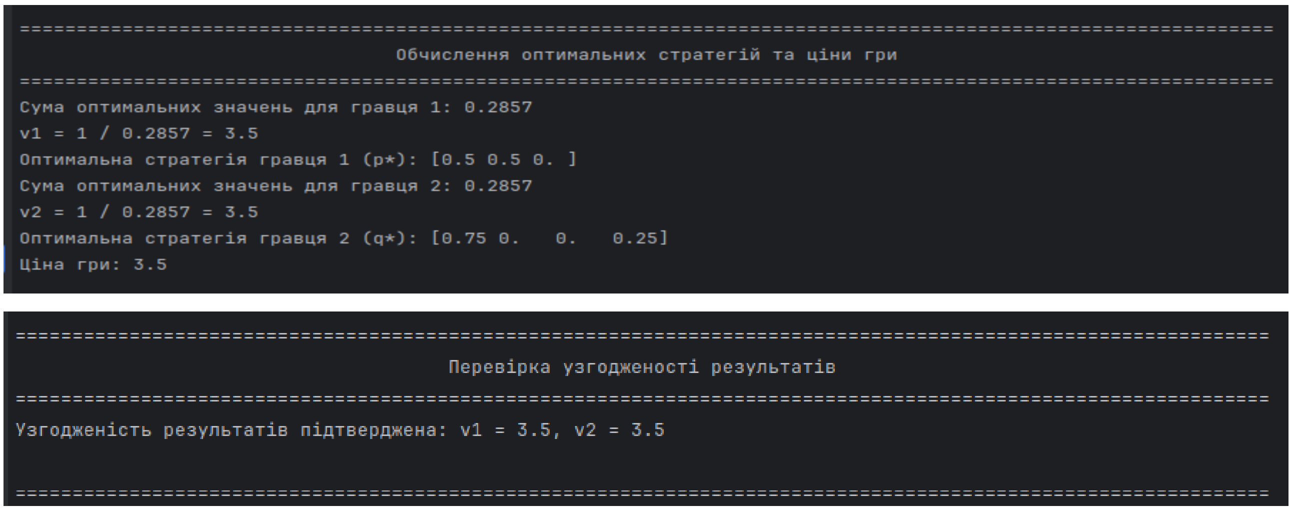


```
==========================================================================================
                          Задача лінійного програмування для гравця 1
==========================================================================================
f(x) =  x1 + x2 + x3  -> min

Система обмежень:
4.0*x1 + 3.0*x2 + 2.0*x3 >= 1
3.0*x1 + 4.0*x2 + 5.0*x3 >= 1
4.0*x1 + 6.0*x2 + 7.0*x3 >= 1
2.0*x1 + 5.0*x2 + 3.0*x3 >= 1
x1, x2, ..., xm >= 0
```

```
==========================================================================================
                    Розв'язання задачі лінійного програмування для гравця 1
==========================================================================================
Оптимальні значення змінних для гравця 1: [0.14285714 0.14285714 0.        ]
Значення цільової функції: 0.2857142857142857

==========================================================================================
                          Задача лінійного програмування для гравця 2
==========================================================================================
f(x) =  y1 + y2 + y3 + y4  -> max

Система обмежень:
4.0*y1 + 3.0*y2 + 4.0*y3 + 2.0*y4 <= 1
3.0*y1 + 4.0*y2 + 6.0*y3 + 5.0*y4 <= 1
2.0*y1 + 5.0*y2 + 7.0*y3 + 3.0*y4 <= 1
y1, y2, ..., yn >= 0

==========================================================================================
                    Розв'язання задачі лінійного програмування для гравця 2
==========================================================================================
Оптимальні значення змінних для гравця 2: [0.21428571 0.         0.         0.07142857]
Значення цільової функції: 0.28571428571428575

==========================================================================================
```

Figure 2 – Screenshot of the program's operation showing the construction and solution of dual linear programming problems for both players

Authoring



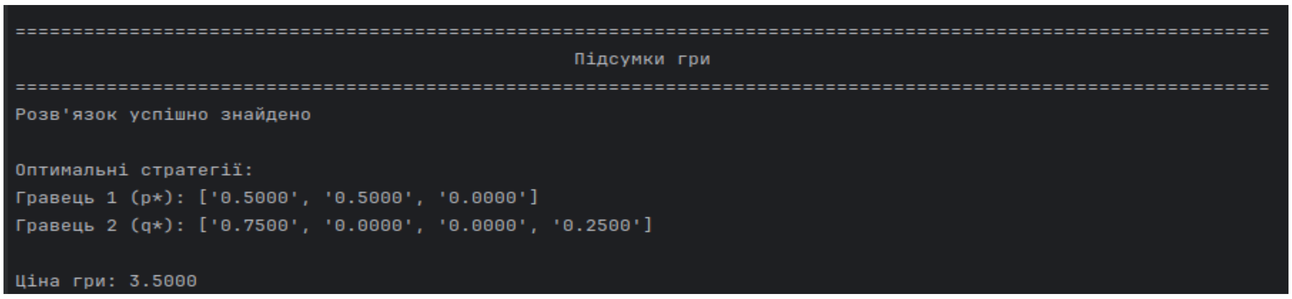
```
==========================================================================================
                          Обчислення оптимальних стратегій та ціни гри
==========================================================================================
Сума оптимальних значень для гравця 1: 0.2857
v1 = 1 / 0.2857 = 3.5
Оптимальна стратегія гравця 1 (p*): [0.5 0.5 0. ]
Сума оптимальних значень для гравця 2: 0.2857
v2 = 1 / 0.2857 = 3.5
Оптимальна стратегія гравця 2 (q*): [0.75 0.   0.   0.25]
Ціна гри: 3.5
```

```
==========================================================================================
                              Перевірка узгодженості результатів
==========================================================================================
Узгодженість результатів підтверджена: v1 = 3.5, v2 = 3.5


==========================================================================================
```

Figure 3 – Screenshot of the program determining the game's solution and checking result consistency

Authoring

Following this, the obtained solutions from the dual problems are used to calculate the players' optimal mixed strategies and the value of the game. A consistency check of these results is performed (Figure 3), and the final outcomes of the matrix game are displayed (Figure 4).



```
==========================================================================================================
                                              Підсумки гри
==========================================================================================================
Розв'язок успішно знайдено

Оптимальні стратегії:
Гравець 1 (p*): ['0.5000', '0.5000', '0.0000']
Гравець 2 (q*): ['0.7500', '0.0000', '0.0000', '0.2500']

Ціна гри: 3.5000
```

Figure 4 – Screenshot of the final game results

Authoring

An analysis of the program's operation revealed the absence of a saddle point in pure strategies (Figure 1), necessitating a solution in mixed strategies. The problem was automatically converted into a pair of dual linear programming problems, which were then solved using the simplex algorithm implemented in our developed software (Figure 2). As a result, Figure 3 shows the optimal mixed strategies $p^* = (0{,}5; 0{,}5; 0)$ and $q^* = (0{,}75; 0; 0; 0{,}25)$ for the first and second players, respectively, along with the game's value $V = 3{,}5$. To confirm the correctness of the obtained solution, optimality conditions were verified, demonstrating that the found strategies are indeed Nash equilibrium strategies. The effectiveness of the developed software was evaluated based on the time required to solve problems of varying dimensions. Comparison with existing software tools showed that the proposed approach significantly reduces solution time for medium and large-sized games.

Capabilities and Practical Applications of the Developed Software.

The developed software is a powerful tool for automated analysis of conflict situations and solving game theory problems using linear programming methods, ensuring a comprehensive approach from data input to interpretation of results. Among its key capabilities are the automatic modeling of conflicts by transforming payoff matrices into formalized linear programming problems, as well as the detection of saddle points and, if necessary, the transformation of matrices with negative values. The program automatically forms a pair of mutually dual linear programming problems for both players and, thanks to the integration of the linprog algorithm from the SciPy library, effectively solves them, accurately calculating optimal mixed or pure strategies for each participant and determining the game's value. This functionality is critically important for rapid response in a dynamic environment, as the obtained results are transformed into mathematically substantiated recommendations regarding the most profitable behavior, allowing for minimized risks and maximized potential gains. Furthermore, the system's ability to quickly recalculate strategies when input data changes makes it indispensable for adaptive planning and modeling various scenarios, allowing for the product's integration into existing decision support systems for automatic generation and evaluation of optimal strategies in real-time. The use of Python and optimized libraries guarantees scalability for handling large-dimension games.

The practical application of this tool is extremely broad. It can be used in economics and business for analyzing competitive strategies and modeling pricing; in military affairs and security for developing operational strategies; in political science and sociology for analyzing election campaigns; in resource management and logistics for optimizing distribution; and also in scientific research and education for studying and demonstrating game theory and linear programming. Thus, this software is a universal tool for supporting decision-making in conflict situations, providing the ability to find optimal strategies and predict the results of interaction between rational participants.

Summary and conclusions.

As a result of this research, an approach to automating the process of investigating conflict situations and solving game theory problems in mixed strategies by reducing them to linear programming problems has been developed and justified. The main results of the work are as follows:

– Have been systematized and generalized the theoretical foundations for modeling conflict situations using game theory methods, particularly when mixed strategies are employed.

– Were formalized the process of reducing matrix games to linear programming problems and developed an algorithm for automatically forming and solving the corresponding optimization problems.

– Has been developed a software product that implements the full cycle of solving game problems: from inputting initial data to interpreting the results.

– Has been conducted an experimental verification of the developed approach on a series of test and practical problems, which confirmed its effectiveness and reliability.

The obtained results have theoretical and practical significance for the development of methods in operations research, system analysis, decision theory, and game theory. The developed software has been utilized as a standalone tool for analyzing conflict situations.

References.

1. Samuel G. O., Ekoko P. O. (2024). Formulation of game model as a linear programming problem using various models. *African Journal of Mathematics and Statistics Studies*, issue 2, vol. 7, pp. 79–95.
DOI: 10.52589/AJMSS-XHFXSZP7

2. Wolford S. (2019). The Politics of the First World War: A Course in Game Theory and International Security. Cambridge: Cambridge University Press. 464 p.
DOI: 10.1017/9781108349956

3. Itchhaporia D. (2022). Game Theory, Health Care, and Economics. *Journal of the American College of Cardiology*, issue 15, vol. 79, pp. 1542–1543.
DOI: 10.1016/j.jacc.2022.03.331

4. Djulbegovic B, Hozo I, Ioannidis J. (2015). Modern health care as a game theory problem: reply. *European Journal of Clinical Investigation*, issue 4, vol. 45, 443 p.
DOI: 10.1111/eci.12414

5. McNamara J. M., Leimar O. (2020). Game Theory in Biology. Oxford: Oxford University Press. 352 p.
DOI: 10.1093/oso/9780198815778.001.0001

6. McNamara J. M. (2013). Towards a richer evolutionary game theory. *Journal of The Royal Society Interface*, issue 88, vol. 10, 20130544.
DOI: 10.1098/rsif.2013.0544

7. Samuelson L. (2016). Game Theory in Economics and Beyond. *Journal of Economic Perspectives,* issue 4, vol. 30, pp. 107–30**.**
DOI: 10.1257/jep.30.4.107

8. Leimar O., McNamara J. M. (2023). Game theory in biology: 50 years and onwards. *Philosophical Transactions B*, issue 1876, vol. 378, 20210509.
DOI: 10.1098/rstb.2021.0509

9. Frahm G. (2019). Rational Choice and Strategic Conflict: The Subjectivistic Approach to Game and Decision Theory. Berlin: De Gruyter, 2019. 340 p.
DOI: 10.4000/oeconomia.8322

10. Maschler M., Solan E., Zamir S. (2013). Game Theory. Cambridge: Cambridge University Press. 352 p.
DOI: 10.1017/CBO9780511794216

Article sent: 05.2025