ФІЗИКО-МАТЕМАТИЧНА ОСВІТА
(PHYSICAL AND MATHEMATICAL EDUCATION)
p-ISSN: 2413-1571, e-ISSN: 2413-158X **2026, 41(2),** **https://doi.org/10.31110/fmo2026.v41i2-01**
UDC: 519.6

ПРОГРАМНА РЕАЛІЗАЦІЯ НА МОВІ C# АЛГОРИТМІВ ОБЧИСЛЕННЯ КОРЕЛЯЦІЇ ПІРСОНА, СПІРМЕНА І КЕНДАЛЛА

Віталій БАЗУРІН ✉
Державний торговельно-економічний університет, Україна
vbazurin@gmail.com
https://orcid.org/0000-0002-6614-4889

Софія БАЗУРІНА
Глухівський міський центр позашкільної освіти, Україна
sofia.bazurina@gmail.com
https://orcid.org/0009-0004-5168-9992

Світлана КОЛЕСНИК
Відокремлений структурний підрозділ
«Глухівський агротехнічний фаховий коледж СНАУ», Україна
ksa081967@gmail.com
https://orcid.org/0009-0005-9510-288X

В'ячеслав КОЛЕСНИК
Відокремлений структурний підрозділ
«Глухівський агротехнічний фаховий коледж СНАУ», Україна
kvv.slava21@gmail.com
https://orcid.org/0009-0006-0245-2811

Ніна ДУЛЕНКО
Відокремлений структурний підрозділ
«Глухівський агротехнічний фаховий коледж СНАУ», Україна
ninadulenko03@gmail.com
https://orcid.org/0009-0006-5151-9808

SOFTWARE IMPLEMENTATION OF ALGORITHMS FOR CALCULATING PEARSON, SPEARMAN AND KENDALL CORRELATIONS IN C#

Vitalii BAZURIN ✉
State University of Trade and Economics, Ukraine
vbazurin@gmail.com
https://orcid.org/0000-0002-6614-4889

Sofia BAZURINA
Hlukhiv City Center for Extracurricular Education, Ukraine
sofia.bazurina@gmail.com
https://orcid.org/0009-0004-5168-9992

Svitlana KOLESNYK
Separate structural unit
"Hlukhiv Agrotechnical Vocational College of SNAU", Ukraine
ksa081967@gmail.com
https://orcid.org/0009-0005-9510-288X

Vyacheslav KOLESNYK
Separate structural unit
"Hlukhiv Agrotechnical Vocational College of SNAU", Ukraine
kvv.slava21@gmail.com
https://orcid.org/0009-0006-0245-2811

Nina DULENKO
Separate structural unit
"Hlukhiv Agrotechnical Vocational College of SNAU", Ukraine
ninadulenko03@gmail.com
https://orcid.org/0009-0006-5151-9808

АНОТАЦІЯ

Формулювання проблеми. *У наукових дослідженнях з соціології, освіти, фізики, хімії та інших наук використовуються методи кореляційного аналізу. За допомогою коефіцієнтів кореляції визначають існування зв'язку між двома і більше вибірками даних.*

Використання коефіцієнтів кореляції надає можливість досліднику з'ясувати, чи існує зв'язок між двома наборами даних, а також характер цього зв'язку: сильний чи слабкий, прямий чи обернений. Завдяки методам кореляції дослідник має дієвий інструмент для підтвердження чи спростування гіпотези.

Важливе місце у застосуванні кореляційного аналізу на практиці відіграють засоби інформаційних технологій. Ці засоби надають можливість зменшити час обробки результатів наукових досліджень у кілька разів. Програмні засоби, які використовуються під час кореляційного аналізу, варіюються у широкому спектрі: до них належать і електронні таблиці MS Excel, і математичний пакет STATISTICA, і мова програмування R. Спільним недоліком зазначених засобів ІТ є те, що дослідник, який їх використовує, повинен мати певний рівень математичної підготовки (STATISTICA), або навички програмування (мова R), або виконувати введення даних вручну (MS Excel). Виникає суперечність між можливостями ІКТ для кореляційного аналізу і тим, як вони застосовуються зараз.

Матеріали і методи. *Під час виконання дослідження ми використали такі методи: аналіз і синтез наукової літератури з кореляційного аналізу, теорії алгоритмів, об'єктно-*

ABSTRACT

Formulation of the problem. *In scientific research in sociology, education, physics, chemistry and other sciences, methods of correlation analysis are used. Correlation coefficients are used to determine the existence of a relationship between two or more data samples. The use of correlation coefficients allows the researcher to determine whether there is a relationship between two data sets and the nature of that relationship: strong or weak, direct or inverse. Thanks to correlation methods, the researcher has an effective tool for confirming or disproving a hypothesis. Information technology tools play an important role in the application of correlation analysis in practice. These tools enable reducing the processing time for scientific research results by several times. Software tools used in correlation analysis vary widely, from MS Excel spreadsheets to STATISTICA and the R programming language. A common drawback of these IT tools is that the researcher using them must have a certain level of mathematical training (STATISTICA), programming skills (R), or manual data entry skills (MS Excel). There is a discrepancy between the capabilities of ICT for correlation analysis and how they are currently used.*

Materials and methods. *During the research, we used the following methods: analysis and synthesis of scientific literature on correlation analysis, algorithm theory, object-oriented programming, and the foundations of scientific research; comparative analysis of algorithms for calculating Pearson, Spearman, and Kendall correlation coefficients; algorithmic*

орієнтованого програмування, основ наукових досліджень; порівняльний аналіз алгоритмів розрахунку коефіцієнтів кореляції Пірсона, Спірмена, Кендалла; алгоритмічний метод для побудови алгоритмів розрахунку кореляції; методи об'єктно-орієнтованого програмування для програмної розроблених алгоритмів; методи кореляції; тестуваннння обчислювальної складності алгоритмів; тестування точності; тестування виняткових ситуацій (пропуск значень, різні розміри вибірок), тестування стабільності.

Результати. *Реалізовано мовою C# алгоритми обчислення кореляції Пірсона, Спірмена і Кендалла. Під час розробки використано компонентно-орієнтований і об'єктно-орієнтований підходи у програмуванні. Алгоритми обчислення кореляції Пірсона, Спірмена і Кендалла представлені мовою C# у вигляді програмної бібліотеки у форматі DLL, яка складається з 5 класів. Розроблені класи і методи для обчислення кореляції Пірсона, Спірмена і Кендалла. Об'єктивною новизною є програмна реалізація алгоритму обчислення кореляції Кендалла для платформи .NET. Для розрахунку вказаних коефіцієнтів кореляції розроблено додаток CorrAnalyzer, до якого підключено створену програмну бібліотеку. Розроблені бібліотеку і додаток доцільно використовувати у педагогічних, психологічних, соціологічних дослідженнях для визначення зв'язку між двома вибірками.*

Висновки. *Методи кореляції застосовуються для визначення зв'язку між двома вибірками. Найбільш часто використовуються кореляції Пірсона, Спірмена і Кендалла. Коефіцієнти кореляції Пірсона, Спірмена і Кендалла відрізняються за принципом і алгоритмом розрахунку. На основі результатів тестування точності, обчислювальної складності і винятків визначено придатність розроблених методів для застосування під час обробки результатів експерименту. Для того, щоб створювана програмна бібліотека являла собою потужний інструмент для визначення зв'язку між вибірками, в ній реалізовані всі три методи кореляції. Бібліотеку для розрахунку кореляції реалізовано у вигляді компонента (файл DLL), який інтегровано у додаток CorrAnalyzer, написаний мовою C#. Додаток CorrAnalyzer призначений для визначення зв'язку між невеликими вибірками (до 1000 значень).*

У перспективі розроблену програмну бібліотеку доцільно доповнити класами для розрахунку кореляції Метьюза і конкордації.

КЛЮЧОВІ СЛОВА: *кореляція; алгоритм; Пірсон; Спірмен; Кендалл; компонентно-орієнтований підхід; програмна бібліотека.*

method for constructing correlation calculation algorithms; object-oriented programming methods for software development of developed algorithms; correlation methods; computational complexity testing; accuracy testing; exception testing (handling missing values, different sample sizes); numerical stability testing.

Results. *Algorithms for calculating Pearson, Spearman, and Kendall correlations have been implemented in the C# language. During development, component-oriented and object-oriented approaches in programming were used. Algorithms for calculating Pearson, Spearman, and Kendall correlations are presented in the C# language in the form of a program library in DLL format, which consists of 5 classes. Classes and methods for calculating Pearson, Spearman, and Kendall correlations have been developed. The objective novelty is the software implementation of the algorithm for calculating Kendall correlations for the .NET platform. To calculate the specified correlation coefficients, the CorrAnalyzer application has been developed, which connects to the program library created. The developed library and application are recommended for use in pedagogical, psychological, and sociological research to determine the relationship between the two samples.*

Conclusions. *Correlation methods are used to determine the relationship between two samples. The most commonly used are Pearson, Spearman, and Kendall correlations. The Pearson, Spearman, and Kendall correlation coefficients differ in their principles and calculation algorithms. Based on testing results for accuracy, computational complexity, and exceptions, the suitability of the developed methods for processing experimental results was determined. To make the software library a powerful tool for determining relationships between samples, all three of the above correlation methods are implemented in it. The library for calculating correlation is implemented as a component (DLL) that is integrated into the CorrAnalyzer application, written in C#. The CorrAnalyzer application is designed to determine the relationship between small samples (up to 1000 values).*

In the future, it is advisable to supplement the developed software library with classes for calculating the Matthews correlation and concordance.

KEYWORDS: *correlation; algorithm; Pearson; Spearman; Kendall; component-oriented approach; software library.*

ДЛЯ ЦИТУВАННЯ: Bazurin V., Bazurina S., Kolesnyk S., Kolesnyk V., Dulenko N. Software implementation of algorithms for calculating Pearson, Spearman and Kendall correlations in C#. *Фізико-математична освіта*, 2026. Том 41. № 2. С. 6-17. https://doi.org/10.31110/fmo2026.v41i2-01.

FOR CITATION: Bazurin, V., Bazurina, S., Kolesnyk, S., Kolesnyk, V., & Dulenko, N. (2026). Software implementation of algorithms for calculating Pearson, Spearman and Kendall correlations in C#. *Physical and Mathematical Education*, *41*(2), 6-17. https://doi.org/10.31110/fmo2026.v41i2-01.

INTRODUCTION

Formulation of the problem. Scientific research in sociology, education, physics, chemistry, and other sciences uses methods of correlation analysis. Correlation coefficients are used to determine the existence of a relationship between two or more data samples. Using correlation coefficients allows the researcher to find out whether there is a relationship between two data sets, as well as the nature of this relationship: strong or weak, direct or inverse. Thanks to correlation methods, the researcher has an effective tool for confirming or disproving a hypothesis. Information technology tools play an important role in the application of correlation analysis in practice.

Existing solutions do not provide the necessary functionality (Math.NET Numerics, ML.NET do not have tools for calculating Kendall correlation), data entry automation (STATISTICA, MathCAD, MATLAB, Accord.NET), and are designed for outdated .NET platforms (Math.NET Numerics, Accord.NET). The proposed solution is aimed at developing a library and integrating it into a software tool that will fully automate data processing and calculate Pearson, Spearman, and Kendall correlations for given samples.

These tools make it possible to reduce the processing time of scientific research results by several times. Software tools used in correlation analysis vary widely, from MS Excel spreadsheets to STATISTICA and the R programming language. A common drawback of these IT tools is that the researcher using them must have a certain level of mathematical training (STATISTICA), programming skills (R), or manual data entry skills (MS Excel). There is a discrepancy between the capabilities of ICT for correlation analysis and how they are currently used.

Existing software tools provide limited functionality (for example, the Math.NET Numerics library provides Pearson and Spearman correlation calculations, and does not calculate Kendall correlation, MS Excel, MatCAD and MATLAB software tools do not provide the necessary automation of correlation calculations), some libraries do not contain modern implementations (the latest version of Math.NET Numerics was released in April 2022, the latest version of Accord.NET was released in November 2017). Implementing correlation calculation algorithms as a library is advisable, since support for existing libraries (Math.NET

Numerics, Pandas) may be discontinued by developers. In addition, when developing a software tool that integrates many functions (including automation of data processing, report generation, etc.), it is advisable to use your own library so that the functionality of the software tool can be expanded in the future by expanding the functionality of the library (for example, calculating Matthews correlation, concordance, etc.). Therefore, the software implementation of Pearson, Spearman, and Kendall correlation algorithms, designed for the modern version of .NET, is relevant.

Analysis of actual research. Scientific sources on the research problem, in our opinion, can be conditionally divided into the following categories: 1) studies that reveal the mathematical foundations of correlation analysis; 2) studies devoted to the use of software tools and programming languages in correlation analysis. Let us analyze the scientific studies of the first group. N.J. Gogtay and U.M. Thatte, in their article (Gogtay & Thatte, 2017), reveal the basic principles of correlation analysis. They determine that correlation can be positive (direct), negative (inverse), or absent. By the strength of the connection, correlation can be strong, weak, or zero. Therefore, correlation can be, for example, strong positive or strong negative, etc. Researchers indicate that during correlation analysis, the following factors must be taken into account: 1) correlation analysis is inappropriate to use when the input data for calculations are the values of the same variable in the same or different time intervals; 2) it is advisable to plot a scatter plot during correlation analysis; 3) an outlier (a value that rarely occurs in the samples) can significantly affect the correlation coefficient; 4) correlation analysis should not be performed if there is a nonlinear relationship between different variables; 5) if the input data contain two separate groups of values that are significantly different, the calculations may indicate a correlation (when in fact there may not be one); 6) the sample size should be calculated beforehand. If the samples are small, the calculations may show the existence of a direct relationship (when in fact this is not the case); 7) if one sample consists of data that are included in the second sample, the calculations may show the existence of a direct relationship. M. Franzece and A. Iuliano (2018), in their study, examine such types of correlation as covariance, Pearson, Spearman, and Kendall correlation coefficients. In their article, the researchers show confidence intervals and graphs of all types of distributions, examples of applying different methods to small samples, and for data analysis, including the use of the R software tool to calculate correlation. S. Senthilnathan (2019), in his article, notes that in software tools that perform correlation analysis, two coefficients are mainly used: Pearson and Spearman. S. Senthilnathan, in the article, shows the principle of interpreting correlation coefficients in scientific research in the social sciences. P. Schober and T. R. Vetter (2020), in their article, demonstrate examples of the application of Pearson correlation and Spearman correlation in medical research.

Cicuttin et al. (2022) in their article demonstrate the results of developing their own correlation method, which is similar to the Pearson correlation coefficient, and also prove the validity of the developed method as a result of studying impulses.

T. J. Cleophas and A. H. Zwinderman (2018) offer their own approach to determining the correlation between two continuous variables, which consists of applying Bayesian linear correlation analysis. R.J. Janse, T. Hoekstra, J. K. J. Jager, C. Zoccali, G. Tripepi, F.W. Dekker, and M. van Diepen, in their article, reveal the main limitations and hidden moments of correlation analysis (Janse et al., 2021). The article considers issues related to correlation: variance and its interpretation, linearity of correlation, range of observations for correlation, non-causality of correlation, agreement between methods, intraclass coefficient, 95% agreement limit, and Bland-Altman plot. J. Chen, G. Wang, and G. B. Giannakis, in their article, propose a new approach in multi-species canonical correlation analysis with graph regularization, as well as a dual approach adapted for cases when the number of data pairs is less than the size of the data vector (Chen et al., 2019). Let us analyze the second group's research in more detail. D. Makowski, M. S. Ben-Shachar, I. Patil, and D. Lüdecke in their article give examples of calculating the following types of correlation: linear (Pearson), rank (Spearman, Kendall), two-weighted average, distance correlation, percent curvature correlation, Shepherd's Pi correlation, point-biserial and biserial correlation, polychoric, tetrachoric, partial, multilevel (Makowski et al., 2020). The article also provides the results of calculating the correlation using a program in the R language. But there is no assessment of their characteristics, such as time complexity and accuracy. The developed application is posted on the resource (https://github.com/easystats/correlation). Disadvantages: the created application is not intended for the .NET platform.

S. Yadav (2018) indicates that two correlation coefficients are most often used: Pearson and Spearman, and also demonstrates graphs depicting different types of relationships. The article also gives examples of calculating correlation coefficients using the SPSS software.

J.-Ch. Yoo and T.H. Han (2009) in their article demonstrate a fast algorithm for calculating the normalized cross-correlation without using the multiplication operation, as well as the results of testing the algorithm on a sample of 100,000 test signals. A correlation calculation algorithm for signal processing is developed.

W.N. Arifin (2018) developed a calculator for calculating sample sizes used in statistical analysis. Using this calculator, scientists can calculate sample size for Cronbach's alpha, intraclass correlation, kappa, and Pearson's correlation coefficient. B. Winter (2019), in his book, provides examples of calculating Pearson's correlation using R. R. Vallat (2018), in his article, describes a Python package called Pingouin, which is designed to automate statistical calculations, including the calculation of repeated measures correlation coefficients and robust correlations. This package is built on the basis of the Pandas library. In our opinion, this package has only one drawback: it is built on the basis of third-party software, support for which may be discontinued by the developers.

The article (Rayhan, 2025) compares the Pearson, Spearman, and Kendall correlation methods based on the following criteria: type (parametric or nonparametric), data type (for which the method is appropriate), and limitations. The article also provides heat maps for the specified methods, generated using the KARL Lab Correlation Tool (developed by the article's authors). This software tool is written in Python using Streamlit, Pandas, NumPy, SciPy, Matplotlib, and Seaborn libraries. Disadvantages: the developed software tool is not focused on the .NET platform, uses third-party software, and is also intended for use in the field of Data Science and business analytics.

O. Derevyanchuk (2025), in his article, describes the software implementation of the Pearson correlation algorithm in Python in the form of the software tool egres_ed_25. The article also presents the results of research into the relationship between the study of different disciplines by university students. The developed software tool uses the third-party library

Matplotlib. The article does not disclose the accuracy and time complexity of the developed algorithm that calculates the Pearson correlation.

Let's analyze the software tools with which you can calculate correlation. These include STATISTICA, SPSS, Microsoft Excel, the Math.NET Numerics library, and the built-in Correlation method of the C# language.

Using STATISTICA and SPSS, you can calculate Pearson and Spearman correlation coefficients. However, these software tools are paid.

Microsoft Excel provides the ability to calculate the Pearson correlation between two samples using the built-in Correl function. Other correlation coefficients are not calculated.

The software tool FinnCorr (Dakılır, 2024) is a powerful tool for financial analysis and includes a function for calculating Pearson correlation. Disadvantage: limited functionality (only calculates Pearson correlation).

The CorrelEstimator library (Mith0304, 2015) provides the ability to calculate Kendall correlation and is designed for use on the .NET platform. Disadvantages: unstable operation, limited functionality.

The Math.NET Numerics library is a powerful tool for performing algebraic and statistical calculations: probability distribution, pseudorandom number generation, matrix operations, complex numbers, linear regression, curve fitting, financial statistics, Monte Carlo method, Fourier transform, and others (Ruegg, 2025).

The Math.NET Numerics library contains functions for calculating Pearson and Spearman correlations, but does not have a function for calculating Kendall correlations. However, in cases where the data distribution is not normal, there are a large number of identical values, and when the sample size is small, it is advisable to use Kendall correlations. The library we have developed provides the calculation of Kendall correlations.

The time complexity of the algorithm for calculating the Pearson correlation is $O(n)$. In this library, the method for determining the Pearson correlation is optimized so that the number of passes through the data array is minimal (usually 2 passes). Therefore, the Correlation.Pearson() method of the Math.NET Numerics library will be efficient for large samples. The algorithm makes two passes through the data arrays: 1. The first one is to find the average values. 2. The second one is to calculate the numerator and denominator. Each pass has a complexity of $O(n)$. All other operations (division, root, multiplication) are $O(1)$. So, the total complexity: $T(n)=O(n)+O(n)+O(1)=O(n)$.

For Spearman correlation, the complexity of the algorithm is calculated as follows: 1) ranking: sorting the array $O(n\backslash log\ n)$; 2) calculating averages and sums: one pass $O(n)$. Total complexity: $T(n)=O(n\ log\ n)+O(n)=O(n\ log\ n)$.

The Accord.NET library is designed for machine learning, but it includes functions for calculating Pearson, Spearman, and Kendall correlations. However, the latest version of Accord.NET was released in 2017, so new versions of the .NET Framework require a new implementation of this library.

Pearson correlation calculation (in Accord.NET library Pearson correlation realized as a structure PearsonCorrelation in Accord.Math.Distances namespace): 1) first, the average values of the arrays are calculated; 2) then one pass is made for the numerator and denominator. Algorithm complexity: Calculating the averages: one pass through the array – $O(n)$, calculating the sums of deviations: one more pass – $O(n)$. Final operations: constant – $O(1)$. Total algorithm complexity: $T(n)=O(n)$.

Accord.net does not have built-in methods for calculating Spearman and Kendall correlations. For these, you must use ranking functions and write the method for calculating the correlation manually (Measures Methods, 2017).

Results of testing built-in methods for calculating correlation of the Math.NET Numerics and Accord.NET libraries. To process 2 samples of 1000000 elements, the Math.NET Numeric library spends 161483 clock cycles, the Accord.NET library – 787329 clock cycles. The result of the Correlation.Pearson(x, y) method of the Math.NET Numeric library is a number, the Pearson correlation coefficient, in the Measures. Using the Correlation(xy) method of the Accord.NET library, the result is a correlation matrix. This matrix should be further processed using other functions to obtain the Pearson correlation coefficient. From the above data, we can conclude that the Math.NET Numerics library is more suitable for use in the application, since it works faster and using the Accord.NET library requires writing additional code. For the Correlation.Spearman(x, y) method of the Math.NET Numeric library, the test results are as follows: for samples of 1000000 values each, the processing time was 6792208 clock cycles.

API development testing. To use the methods of the Math.NET Numerics library, it is enough to connect the library and call the corresponding method, so according to the "API development" criterion, this library can be assessed at a high level. To use the methods of the Accord.NET library, you need to connect the library itself and write additional code to get the result (for the Pearson or Spearman criterion) in the form of a number, not a matrix. Therefore, according to the "API development" criterion, the Accord.NET library is assessed at an average level.

The analyzed articles contain a description of correlation methods (Pearson, Spearman and Kendall), a comparative analysis of Pearson, Spearman and Kendall correlation methods, a description of the software implementation of correlation methods, a description of the developed applications and their application to determine the accuracy of some correlation methods (mainly Pearson and those developed by the authors), The comparative analysis of Pearson, Spearman and Kendall correlation methods is relative, the authors do not evaluate the developed algorithms using clear metrics (time complexity of the algorithms and their accuracy). The applications developed by the authors are mostly written in Python and use third-party libraries. That is why we consider it advisable to develop our own library for calculating correlation, which will calculate the Pearson, Spearman, and Kendall correlation coefficients.

The purpose of the article is to reveal the features of the software implementation of algorithms for calculating Pearson, Spearman, and Kendall correlations in the C# language as a separate component.

METHODS OF THE RESEARCH

During the research, we used the following methods:

– analysis and synthesis of scientific literature on correlation analysis, algorithm theory, object-oriented programming, and the foundations of scientific research;

– comparative analysis of algorithms for calculating Pearson, Spearman, and Kendall correlation coefficients;
– algorithmic method for constructing correlation calculation algorithms;
– object-oriented programming methods for software development of developed algorithms;
– correlation methods;
- computational complexity testing;
- accuracy testing;
- exception testing (handling missing values, different sample sizes);
- numerical stability testing;

The developed algorithms are implemented in C# for the .NET 4.7 platform in the Microsoft Visual Studio 2022 environment.

Testing the computational complexity of all three developed methods was carried out using built-in methods of C Sharp. During testing, we determined the time complexity of the algorithm for arrays of 1000, 2000, 4000 and 8000 random numbers. During testing, clock cycles were counted. Testing was carried out 5 times, and the average number of cycles required to process the array was determined.

Methodology for testing the accuracy of calculations performed by the algorithm:

1. Calculation of Pearson correlation using MS Excel for arrays of 100, 500 and 1000 numbers.
2. Calculation of Pearson correlation performed by the developed method.
3. Comparison of the obtained values using MS Excel and the developed method.

Testing the accuracy of the Spearman correlation algorithm was carried out for the following sample sizes: 100, 500, and 1000 random numbers. The result was then compared with calculations performed in MS Excel.

RESULTS OF RESEARCH

Let us first consider the algorithm for calculating the Pearson correlation coefficient. It is described in (Cleophas & Zwinderman, 2018) and is calculated using the formula:

$$r_{xy} = \frac{n\cdot\sum_i(x_i\cdot y_i)-(\sum_i x_i)\cdot(\sum_i y_i)}{\sqrt{(n\cdot\sum_i x_i^2-(\sum_i x_i)^2)\cdot(n\cdot\sum_i y_i^2-(\sum_i y_i)^2)}} \quad (1)$$

Where x_i, y_i are the values of both samples.

Fig. 1 shows the algorithm for calculating Pearson's correlation. The main features of the algorithm:

1) to calculate the Pearson's correlation coefficient, a table of values is required (i.e., a two-dimensional array);
2) the first two columns of the array are filled with values, and the other three are calculated;
3) before entering data into the array, they are checked for gaps and symbolic values. If these bugs are present, the input of the values of x and y is skipped;
4) as new values of x_i, y_i are added to the array, their sums, sums of squares, sums of products are calculated;
5) the data contained in the cells of the array are fixed;
6) after the data entry is completed, the Pearson correlation is calculated;
7) it is possible to combine the calculation of the Pearson correlation and the class constructor.

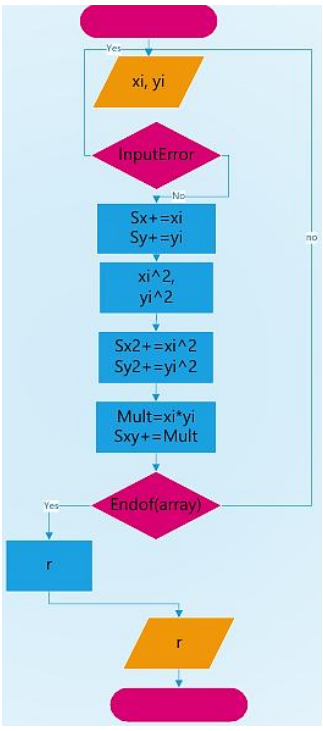

Fig.1. Algorithm for calculating Pearson's correlation
Source: author's development

The *PearsArray* class has been developed to calculate Pearson's correlation. Class members: *RecordsArray* fields (two-dimensional array of values), *N* (number of rows), *Sx, Sy* (sums of samples x and y, respectively), *Sx2* and *Sy2* (sums of squares of samples *x* and *y*), *Sxy* (sum of products *xy*), *r* (correlation coefficient). Class methods: *PearsArray* (constructor), *SetArray* (setting array values), *Correlation* (returns correlation coefficient), *Conclusion* (determines and returns conclusion).

Let us consider the results of testing the developed method for calculating Pearson correlation. The results of the time-complexity test are shown in Fig. 2.

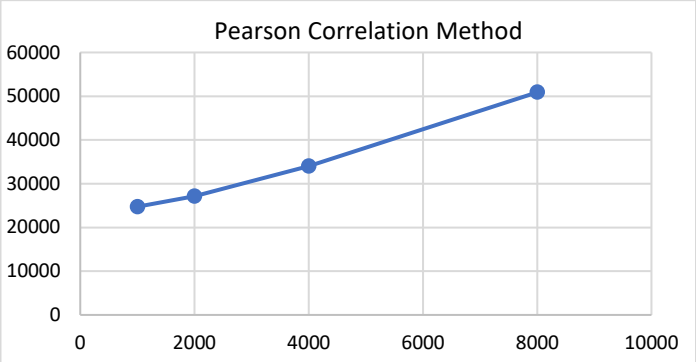

Fig.2. Results of testing the time complexity of the algorithm for calculating the Pearson correlation method

Source: author's research

The figure shows that the time complexity of the algorithm *O* is directly proportional to the sample size *(O(n))*.

Results of testing the accuracy of the developed method for calculating Pearson correlation. The algorithm for calculating Pearson's correlation was tested for accuracy on samples of 100, 500, and 1000, and the results were compared with those calculated in MS Excel.

Table 1. Results of testing the accuracy of calculations using the Pearson correlation method

Sample size	Calculation results of the method	Calculation results in MS Excel	Absolute error
100	-0.118945878	-0.118945878	0
500	0.009701465	0.009701465	0
1000	-0.024792625	-0.024792625	0

Source: author's research

Therefore, the developed algorithm achieves a calculation accuracy of up to 0.000000001.

Testing for missing values and unequal samples was performed during the creation of the *PearsArray* class instance. If a missing value was detected in one of the samples or if the samples had different lengths, an exception was thrown in the *PearsArray* class constructor and the creation of the class instance was stopped.

Let's consider the algorithm for calculating Spearman's correlation and its software implementation.

The sequence of calculating Spearman's correlation is disclosed in (Gogtay & Thatte, 2017). Spearman's correlation is calculated by the formula:

$$r_s = 1 - \frac{6 \cdot \sum (R(x)_i - R(y)_i)^2}{n \cdot (n^2 - 1)} \quad (2)$$

A feature of Spearman's correlation is that it calculates the ranks of the sample x and the sample y separately. Features of the algorithm:

1) the data for calculations can be presented in the form of a two-dimensional array;

2) the first two columns of the array are filled with the initial data, the other three columns of the array contain the calculated data;

3) to determine the rank of each value from the samples x and y, it is necessary to make a pass through the array;

4) after calculating the ranks, another pass through the array should be made to calculate the square of the difference in ranks, the sum of the ranks, and the sum of the squares of the ranks;

5) in the next step, the correlation coefficient is calculated;

6) each row of the table should be implemented as an instance of the *SpRecord* class; the correctness of the entered data is checked in the class constructor; if the entered non-numeric value or the value is omitted, an exception is generated, and the algorithm proceeds to the next step;

7) It is advisable to organize the entire data table as an array of *SpRecord* class objects.

To calculate Spearman's correlation, the *SpRecord* class (related sample values) and the *SpArray* class (contains all data for calculations) were created.

The SpRecord class has the fields *valX, valY* (related sample values), *RankX* and *RankY* (respectively, the ranks of the *x* and *y* values), *dif2* (the square of the difference in ranks of x and y). Class methods: constructor.

The *SpArray* class is an array of *SpRecord* records and has the following fields: *SpRecord* (array of records), *N* (number of records), *Sx* and *Sy* (sum of ranks of the first and second samples), *Sdif* (sum of square differences), *r* (correlation coefficient), *concl* (conclusion). *SpRecord* class methods: *SpArray* (constructor), *AddRecord* (add record to array), *Correlation* (returns correlation coefficient), *Conclusion* (returns conclusion about the existence of relationships).

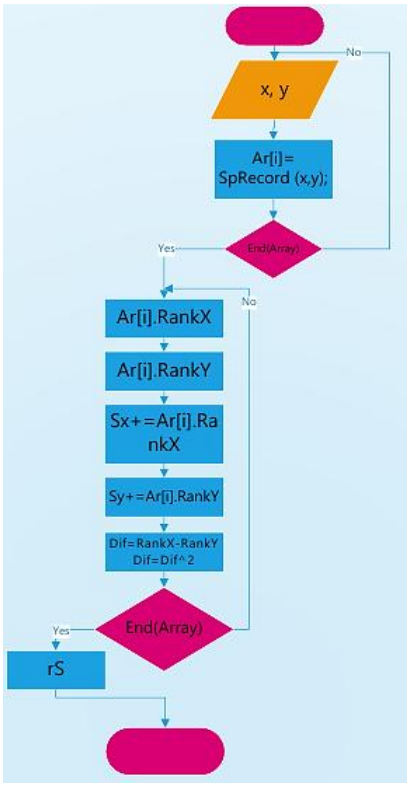

Fig.3. Algorithm for calculating Spearman's correlation
Source: author's development

The results of testing the time complexity of the algorithm for calculating the Spearman correlation are shown in Fig. 4. The figure shows that the algorithm's time complexity is proportional to the square of the sample size *(O(n^2))*. The graph of the *O* function has the form of a parabola.

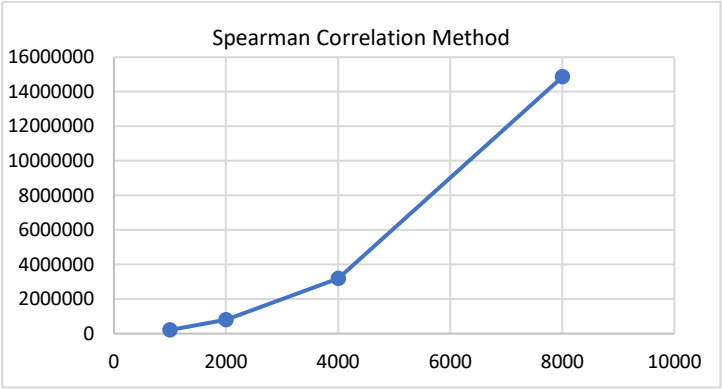

Fig. 4. Results of testing the computational complexity of the algorithm for calculating the Spearman correlation
Source: author's development

The results of testing the accuracy of the developed algorithm for determining the Spearman correlation coefficient are given in Table 2.

Table 2. Results of testing the accuracy of calculations using the Spearman correlation method

Sample size	Calculation results of the method	Calculation results in MS Excel	Absolute error
100	0.058919892	0,058919892	0
500	0.05650007	0.05650007	0
1000	0.04927515	0.04927515	0

Source: author's research

Therefore, the developed algorithm provides calculation accuracy up to 0.00000001.

Testing for missing values and unequal samples was performed during the creation of the *SpArray* class instance. If a missing value was detected in one of the samples or if the samples had different lengths, an exception was thrown in the *SpArray* class constructor, and the creation of the class instance was stopped.

Let's consider the algorithm for calculating the Kendall correlation and its software implementation.

The sequence for calculating the Kendall correlation is given in (Franzece & Iuliano, 2018). The Kendall correlation coefficient is calculated using the formula:

$$\tau = \frac{\sum P - \sum Q}{n \cdot (n-1)/2} \quad (3)$$

Features of the algorithm for calculating the Kendall correlation coefficient:

1) a table of values is used for calculations, in which the elements of the samples x and y are linked to each other; each pair of such linked values is combined into an instance of the *KendRecord* class; valid values are checked in the *KendRecord* class constructor; in case of incorrect data, an exception is generated, in this case an instance of the *KendRecord* class is not created;

2) the table has 6 columns: the values of the samples *x* and *y* are entered in the first 2 columns, the next two columns are the ranks of the values *x* and *y*, respectively, the last two columns are the number of coincidences and inversions, respectively;

3) during calculations, the rows of the table are sorted in ascending order (by the rank of the sample *x*, and the linked values of the sample y must be moved);

4) subsequently, for each value of the sample *y*, the number of coincidences and inversions is calculated.

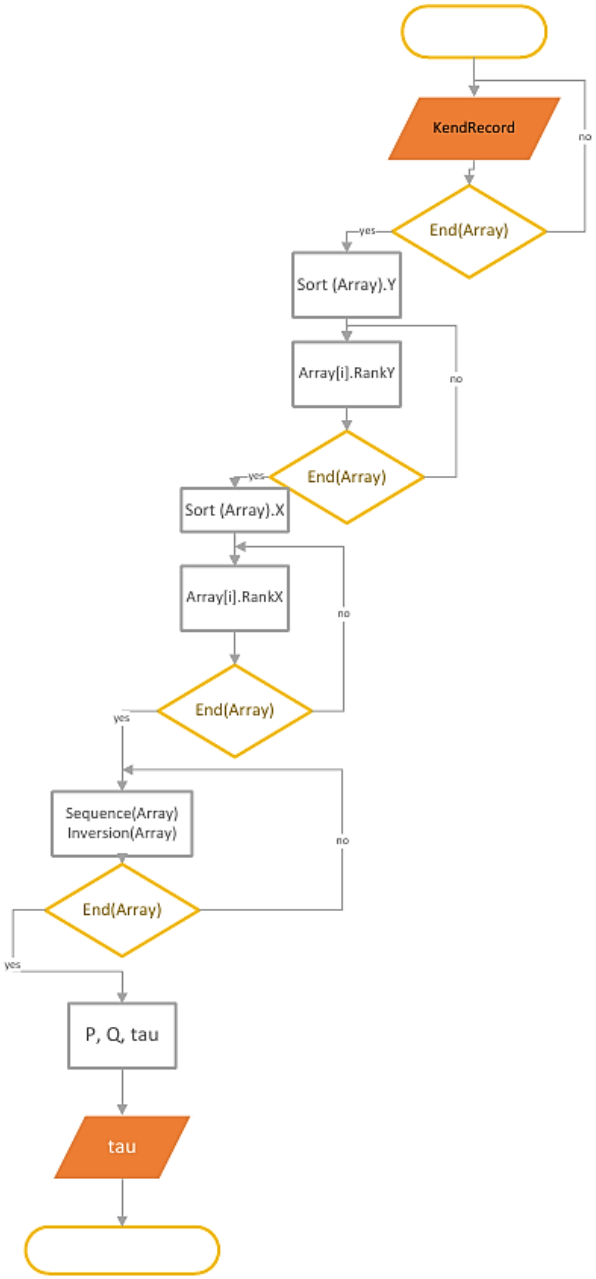

Fig. 5. Schematic of the algorithm for calculating the Kendall correlation coefficient

Source: author's development

To calculate the Kendall correlation, the *KendRecord* and *KendArray* classes were developed. *KendRecord* is a linked pair of values from both samples and has the following fields: *valX* and *valY* (linked values of both samples), *RankX* and *RankY* (ranks of x and y values), *P* (number of matches), *Q* (number of inversions). Class methods: *KendRecord* (constructor).

The *KendArray* class is an array of records (objects of the *KendRecord* class) and contains the following fields: records (array of values), *N* (number of elements in each sample), *P* (sum of matches), *Q* (sum of inversions), *tau* (Kendall correlation coefficient), *concl* (conclusion). Class methods: *KendRecord* (constructor), *AddRecord* (adds a record to the array), *Correlation* (returns the correlation coefficient), *Conclusion* (returns the conclusion).

The results of testing the time complexity of the algorithm for calculating the Kendall correlation method are shown in Fig. 6. The figure shows that the time complexity of the algorithm is proportional to the square of the sample size *(O(n^2)).* The graph of the *O* function has the form of a parabola.

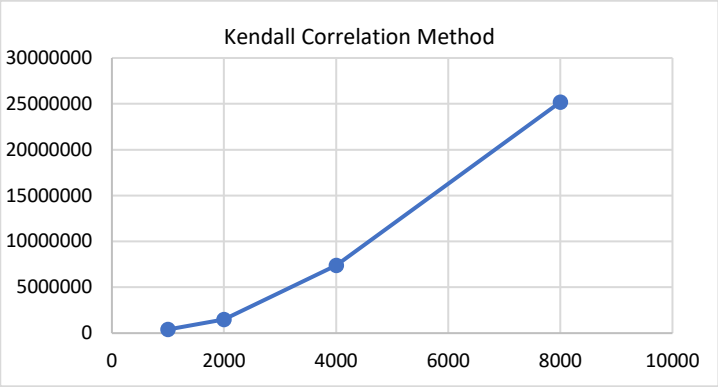

Fig. 6. Results of testing the computational complexity of the algorithm for calculating the Kendall correlation method
Source: author's development

The results of testing the accuracy of the developed algorithm for determining the Spearman correlation coefficient are given in Table 3.

Table 3. Results of testing the accuracy of calculations using the Kendall correlation method

Sample size	Calculation results of the method	Calculation results in MS Excel	Absolute error
100	-0.655959	-0.655959	0
500	0.591991984	0.591991984	0
1000	0.391101101	0.391101101	0

Source: author's research

Therefore, the developed algorithm provides calculation accuracy up to 0.00000001.

Testing for missing values and unequal samples was performed during the creation of the *KendArray* class instance. If a missing value was detected in one of the samples or if the samples had different lengths, an exception was thrown in the *KendArray* class constructor, and the creation of the class instance was stopped.

All developed classes are arranged in one component (module) and implemented in C#. The class diagram of the *CSCorrelation* library is shown in Fig. 7. The members of each class are described earlier.

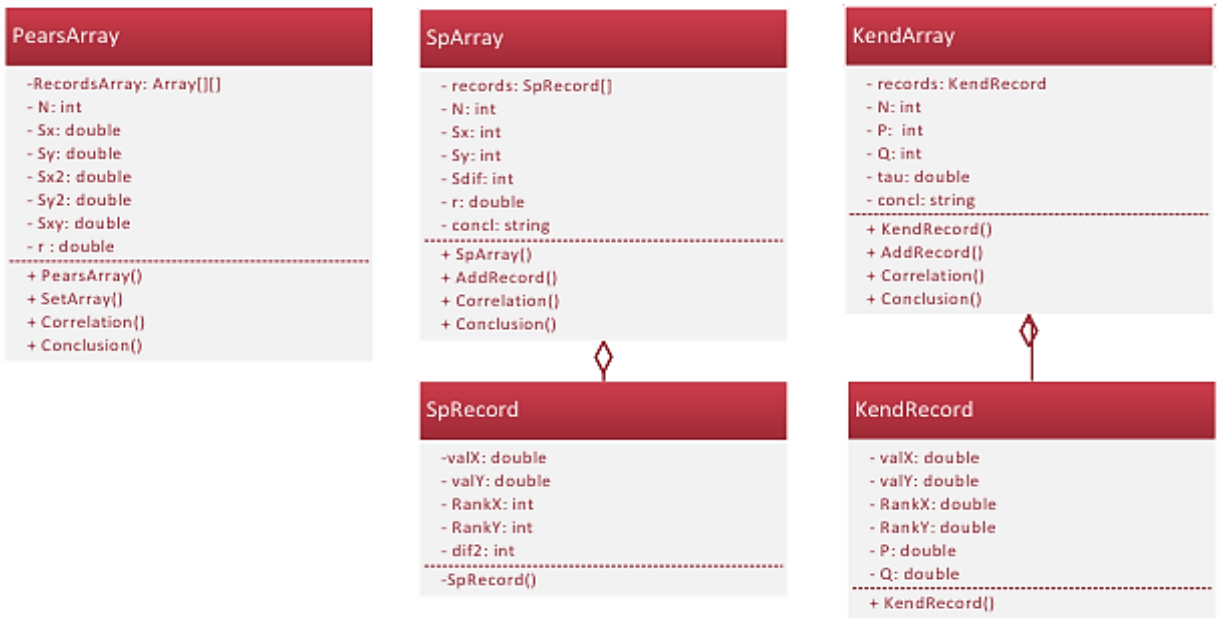

Fig. 7. Contents of the CSCorrelation library
Source: author's development

The developed DLL library is connected to the CorrAnalyzer application (Fig. 8).

The application for calculating correlations consists of a main form and a secondary form. The main form displays a calculation table. This table has a different appearance for Pearson correlation (Fig. 8), Spearman correlation (Fig. 9), and Kendall correlation (Fig. 10).

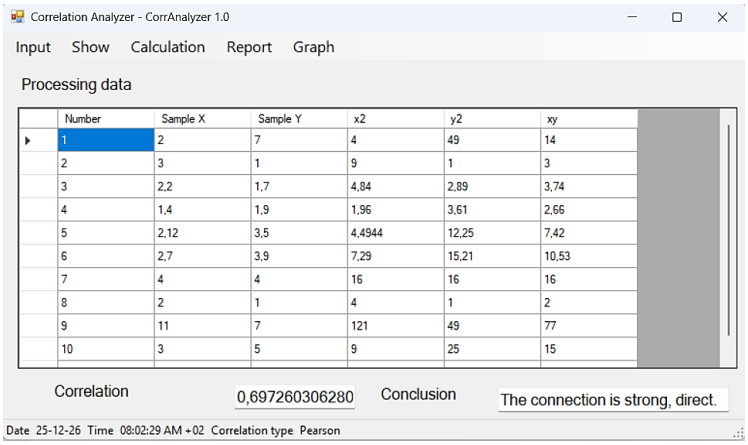

Fig. 8. CorrAnalyzer application (calculation of Pearson correlation)

Source: author's development

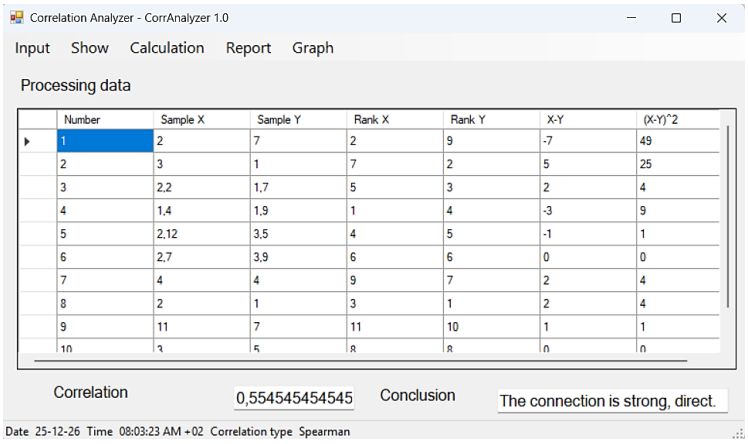

Fig. 9. CorrAnalyzer application (calculation of Spearman correlation)

Source: author's development

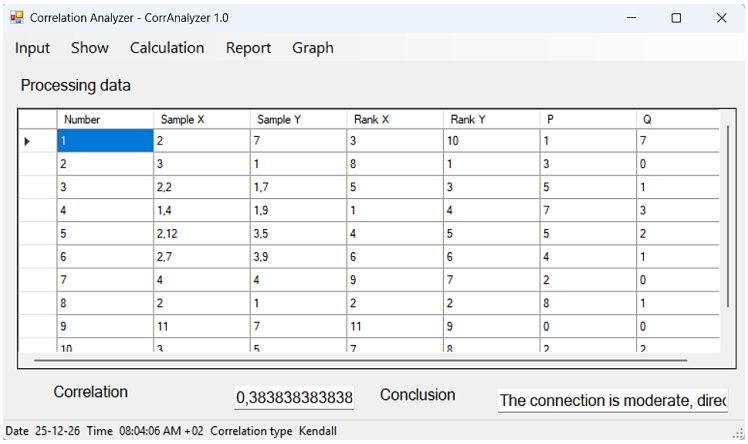

Fig. 10. CorrAnalyzer application (Kendall correlation calculation)

Source: author's development

The additional form displays a frequency graph to determine the type of distribution of sample values: normal, exponential, etc. (Fig. 11).

To check the accuracy of calculations in MS Excel, we created a random data set consisting of 2 samples of 11 values each. The calculated values of Pearson's, Spearman's, and Kendall's correlations are 0.695, 0.556, and 0.384, respectively. For the CorrAnalyzer application, the calculated value of Pearson's correlation for this sample is 0.697, Spearman's correlation is 0.554, and Kendall's correlation is 0.383. Thus, the accuracy of the calculations of this application is up to the third decimal place. It follows that the created application is advisable to use for relatively small samples (up to 1000 values).

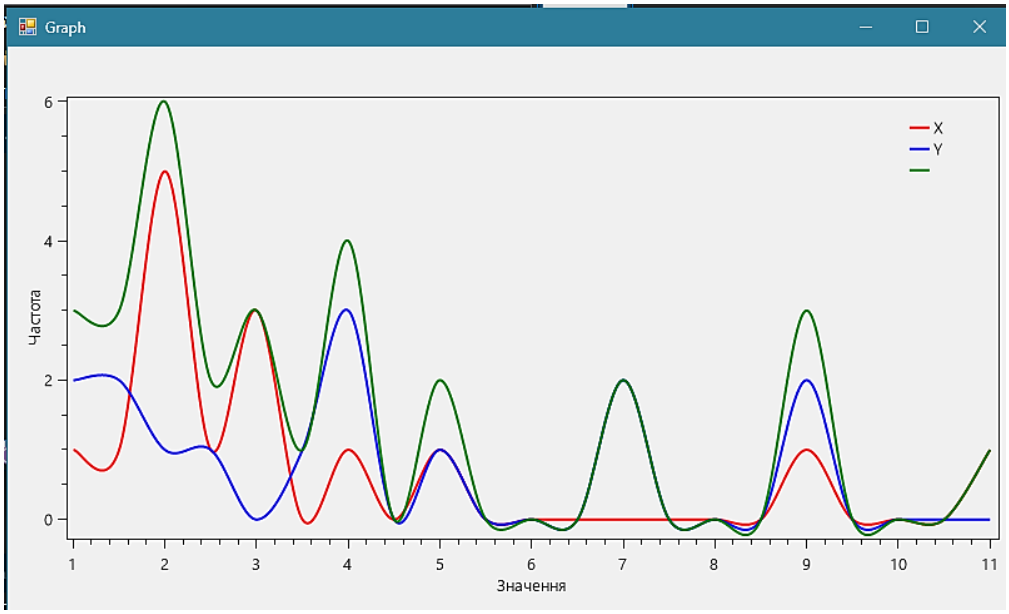

Fig. 11. CorrAnalyzer application (frequency graph)
Source: author's development

DISCUSSION

When developing a software tool for the .NET platform that calculates correlation, you can use the free .NET Numerics and Accord libraries. But their functionality is limited, so when developing such a program, you need to write your own methods, for example, to calculate the Kendall correlation. The approach proposed in the article involves developing your own library for calculating Pearson, Spearman, and Kendall correlations and integrating them into one library. The proposed approach is more rational from the perspective of further development of software tools, as it enables the addition of methods for calculating concordance, Matthews correlation, etc., thereby significantly expanding the functionality of the developed library.

CONCLUSIONS AND PERSPECTIVES FOR FURTHER RESEARCH

Correlation methods are used to determine the relationship between two samples. The most commonly used are Pearson, Spearman, and Kendall correlations. The Pearson, Spearman, and Kendall correlation coefficients differ in their principles and calculation algorithms. To make the software library a powerful tool for determining relationships between samples, all three of the above correlation methods are implemented in it. The library for calculating correlation is implemented as a component (DLL) that is integrated into the CorrAnalyzer application, written in C#. The CorrAnalyzer application is designed to determine the relationship between small samples (up to 1000 values).

During the research, software implementations of algorithms for calculating Pearson, Spearman, and Kendall correlations were developed as a library for the .NET platform. The authors' contribution is a software implementation of the algorithm for calculating Kendall correlations for the .NET platform, since the most common .NET libraries, Numerics and Accord, do not have such a function. The developed methods for calculating Pearson, Spearman, and Kendall correlations were tested for time complexity, accuracy, and failure rates (due to missing values and unequal sample sizes). All methods demonstrated sufficient accuracy (the calculations were checked in MS Excel). The time complexity of the developed methods for calculating Pearson and Spearman correlations is of the same order as the corresponding methods of the .NET Numerics and Accord libraries. The time complexity of the algorithm for calculating the Kendall correlation is *O(n^2)*, since it uses ranking. Research limitations: testing of computational accuracy was limited to sample sizes of 100, 500, and 1000 random numbers, since the library under development is intended for processing the results of pedagogical research, and the number of values is limited to these sizes.

In the future, it is advisable to test the developed algorithm for calculating the Kendall correlation on samples of sizes 2000, 4000, and larger, using more random numbers.

In the future, it is advisable to supplement the developed software library with classes for calculating the Matthews correlation coefficient and the concordance.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

FUNDING SOURCES

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

DATA AVAILABILITY

This study did not generate or utilize any separate datasets.

USE OF ARTIFICIAL INTELLIGENCE (AI) TOOLS

No AI-assisted technologies were used in the preparation of this manuscript.

AUTHOR CONTRIBUTIONS (CRediT)

Vitalii Bazurin – Conceptualization; Methodology; Project administration; Writing – original draft; Writing – review & editing.

Sofia Bazurina – Data curation; Formal analysis; Visualization; Writing – review & editing.

Svitlana Kolesnyk – Validation; Supervision; Writing – review & editing.

Vyacheslav Kolesnyk – Data curation; Formal analysis; Vizualization.

Nina Dulenko – Data Curation; Formal analysis.

REFERENCES (TRANSLATED AND TRANSLITERATED)

1. Accord.NET Framework (n.d.). *Measures.Correlation Method (Double[,]).* URL: http://accord-framework.net/docs/html/M_Accord_Statistics_Measures_Correlation.htm
2. Arifin, W. N. (2018). A Web-based Sample Size Calculator for Reliability Studies. *Education in medicine journal*, *10*(3), 67–76. https://doi.org/10.21315/eimj2018.10.3.8
3. Chen, J., Wang, G., & Giannakis, G. B. (2019). Graph multiview canonical correlation analysis. *IEEE Transactions on Signal Processing*, *67*(11), 2826-2838. https://doi.org/10.1109/TSP.2019.2910475
4. Cicuttin, A., Morales, I. R., Crespo, M. L., Carrato, S., García, L. G., Molina, R. S., Valinoti, B., & Folla Kamdem, J. (2022). A Simplified Correlation Index for Fast Real-Time Pulse Shape Recognition, *Sensors*, *22*(20), 7697. https://doi.org/10.3390/s22207697
5. Cleophas, T. J., & Zwinderman, A. H. (2018). Bayesian Pearson correlation analysis. In *Modern Bayesian statistics in clinical research* (pp. 111-118). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-92747-3_11
6. Dakılır, D. (2024). *FinnCorr: Financial Data Correlation Analysis Engine* (Computer software). GitHub. URL: https://github.com/zelosleone/FinnCorr
7. Derevyanchuk, O. (2025). Methods of correlation and regression of educational data in the systematic analysis of the quality of training of teachers of vocational education. *Pedagogical Academy: scientific notes, 23*. https://doi.org/10.5281/zenodo.18004752
8. Franzece, M., & Iuliano, A. (2018). Correlation analysis. In *Encyclopedia of bioinformatics and computational biology: ABC of bioinformatics* (Vol. 1, pp. 706-721). Elsevier. https://doi.org/10.1016/B978-0-12-809633-8.20358-0
9. Gogtay, N. J., & Thatte, U. M. (2017). Principles of correlation analysis. *Journal of the Association of Physicians of India*, *65*(3), 78-81. https://www.kem.edu/wp-content/uploads/2012/06/9-Principles_of_correlation-1.pdf
10. Janse, R. J., Hoekstra, T., Jager, K. J., Zoccali, C., Tripepi, G., Dekker, F. W., & Van Diepen, M. (2021). Conducting correlation analysis: important limitations and pitfalls. *Clinical Kidney Journal*, *14*(11), 2332-2337. https://doi.org/10.1093/ckj/sfab085
11. Makowski, D., Ben-Shachar, M. S., Patil, I., & Lüdecke, D. (2020). Methods and algorithms for correlation analysis in R. *Journal of Open Source Software*, *5*(51), 2306. https://doi.org/10.21105/joss.02306
12. MathNet.Numerics (n.d.). Correlation – Math.NET Numerics Documentation. URL: https://numerics.mathdotnet.com/api/MathNet.Numerics.Statistics/Correlation.htm#Pearson
13. MathNet.Numerics (n.d.). Correlation – Math.NET Numerics Documentation. URL: https://numerics.mathdotnet.com/api/MathNet.Numerics.Statistics/Correlation.htm#Spearman
14. Measures Methods. (2017). URL: http://accord-framework.net/docs/html/Methods_T_Accord_Statistics_Measures.htm
15. Mith0304. (2015). *CorrelEstimator* (Computer software). GitHub. URL: https://github.com/Mith0304/CorrelEstimator
16. Rayhan, M., Al, A., Md Nurnabe Sagor, Pranto Das, Md. Sabbir Ahmed, Abu Sadat, Abdul Hafiz Tamim, Emon, S., Asad, M. A., & Alam, M. K. (2025). *An Open-Source Framework for Advanced Correlation Analysis: The KARL Lab Correlation Tool (Pro Edition)*. Zenodo. https://doi.org/10.5281/zenodo.17088075
17. Ruegg, C. (2025). Math.NET Numerics (Source code). GitHub. URL: https://github.com/mathnet/mathnet-numerics
18. Schober, P., & Vetter, T. R. (2020). Correlation analysis in medical research. *Anesthesia & Analgesia*, *130*(2), 332. https://doi.org/10.1213/ANE.0000000000004578
19. Senthilnathan, S. (2019). Usefulness of correlation analysis. *Available at SSRN 3416918*. http://doi.org/10.2139/ssrn.3416918
20. StatisticFormula.Correlation (2025). URL: https://learn.microsoft.com/en-us/dotnet/api/system.web.ui.datavisualization.charting.statisticformula.correlation?view=netframework-4.8.1
21. Vallat, R. (2018). Pingouin: statistics in Python. *J. Open Source Softw.*, *3*(31), 1026. http://doi.org/10.21105/joss.01026
22. Winter, B. (2019). *Statistics for linguists: An introduction using R*. Routledge. URL: https://www.jeffreyheinz.net/classes/20F/materials/readings/Bodo2020-FrontMatter.pdf
23. Yadav, S. (2018). Correlation analysis in biological studies. *Journal of the Practice of cardiovascular sciences*, *4*(2), 116-121. https://doi.org/10.4103/jpcs.jpcs_31_18
24. Yoo, J.-Ch., & Han, T.H. (2009). Fast Normalized Cross-Correlation. *Circuits, Systems, and Signal Processing*, *28*(6), 819-843. https://doi.org/10.1007/s00034-009-9130-7

| *Received: 05.01.2026* | *Accepted: 15.03.2026* | *Published: 30.04.2026* |