Бобер Д. О., студент, Сергієнко Г. Є., студент, Лагода О. А., доцент
Київський національний університет технологій та дизайну

АНАЛІЗ ПОХИБОК ПРИ ВИКОРИСТАННІ ЛОКАЛЬНОЇ ФУНКЦІЇ МУАВРА-ЛАПЛАСА ДЛЯ НАБЛИЖЕННЯ ФОРМУЛИ БЕРНУЛЛІ

Анотація. *Унаочнено наближення формули Бернуллі за допомогою локальної функції Муавра-Лапласа шляхом написання програм на Rust та C#. Візуалізовано отримані результати, проаналізовано похибки виконаних наближень та час виконання програм, написаних різними мовами.*

Ключові слова: *схема Бернуллі, формула Бернуллі, локальна теорема Муавра-Лапласа, похибка наближення, інтерфейс, швидкість обчислень.*

Bober D., Sergiienko G., Lagoda O.
Kyiv National University of Technologies and Design

ERROR ANALYSIS IN APPROXIMATION OF THE BERNOULLI FORMULA WITH THE LOCAL MOIVRE-LAPLACE FUNCTION

Abstract. *The approximation of Bernoulli's formula using the local Moivre-Laplace function is demonstrated by writing programs in Rust and C#. The results are visualized, the errors of the approximations and the execution time of programs written in different languages are analyzed.*

Keywords: *Bernoulli scheme, Bernoulli formula, local Moivre-Laplace theorem, approximation error, interface, computation speed.*

Вступ. В науковій та практичній діяльності доводиться мати справу з випробуваннями, які повторюються за схожих умов. Якщо в кожному з випробувань деяка подія А може з’явитися з однією і тією ж ймовірністю *p*, незалежно від результатів попередніх та подальших випробувань, то така схема була вперше досліджена швейцарським вченим Якобом Бернуллі в «Ars conjectandi» і носить його ім’я. Закономірності, що були відкриті при вивченні послідовних незалежних випробувань, сприяли появі не лише формули Бернуллі, а й локальної та інтегральної теорем Муавра-Лапласа та закону великих чисел [1–4]. Таким чином були знайдені ймовірності $P_n(k)$ та $P_n(k_1, k_2)$ – ймовірності того, в серії з n незалежних випробувань, які задовольняють умови схеми Бернуллі, подія А відбудеться рівно k разів та від k_1 до k_2 разів відповідно. Також отримано, що теорема Муавра-Лапласа описує наближення нормального розподілу до біноміального та є окремим випадком центральної граничної теореми [2, 3]. Вищезгадані теореми розглядаються при вивченні дисципліни «Теорія ймовірностей та математична статистика» студентами інженерних спеціальностей в університетах [5–8]. За інерцією звучить, що при великих *n* та *k* обчислення ймовірності $P_n(k)$ за формулою Бернуллі технічно складне. [2] Навіть для студентів-другокурсників очевидно, що ця фраза вже не така актуальна як триста, сто та й навіть 30 років тому. За нинішніх технічних можливостей виникла ідея замість використання табульованих значень локальної функції Муавра-Лапласа, що є вкрай незручним через труднощі з пошуком та якість самої таблиці, чи застосування Excel [9], який передбачає наявність комп’ютера, запрограмувати різними мовами знаходження ймовірностей за допомогою формули Бернуллі та локальної теореми Муавра-Лапласа, порівняти час та точність обчислень і візуалізувати отримані результати у вигляді сайту чи додатку.

Постановка завдання. Використовуючи Rust, C# та Python для написання коду, унаочнити наближення формули Бернуллі за допомогою локальної функції Муавра-Лапласа. Візуалізувати отримані результати, проаналізувати похибки виконаних наближень та порівняти час виконання програм написаних різними мовами.

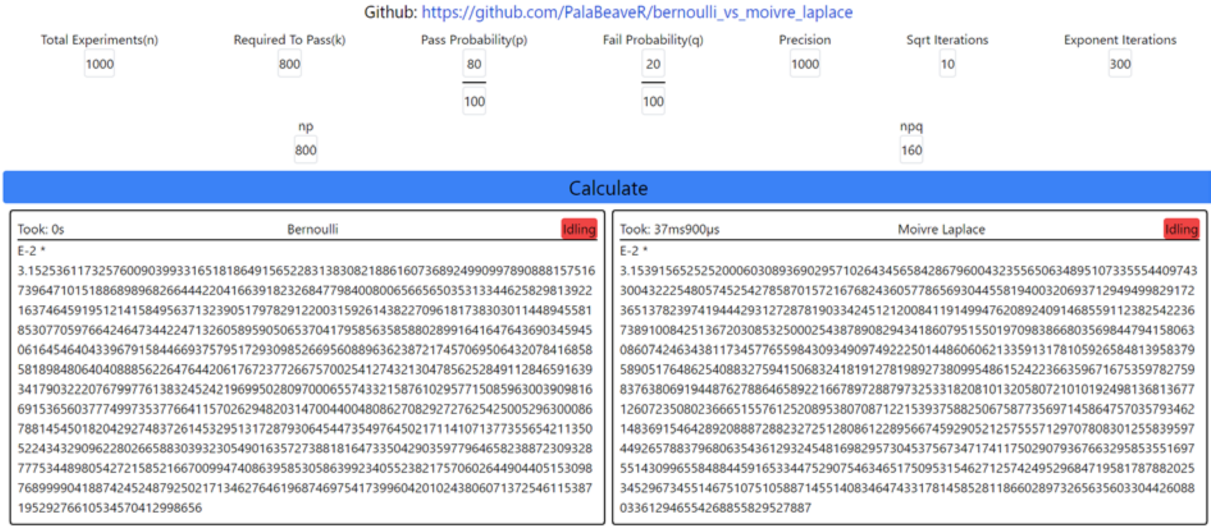
Результати досліджень. Коди було реалізовано мовами Rust, C#.

Ключові моменти коду для формули Бернуллі та Муавра-Лапласа на Rust:

```rust
pub fn bernoulli(
    experiments: u32,
    positive_outcomes: u32,
    positive_probability: FR,
) -> SolverResult {
    let now = Instant::now();

    let (positive_numer, prob_denom) =
        positive_probability.into();

    let negative_numer = &prob_denom - &positive_numer;

    // Picking the biggest of the two factorials in the denumenator of combinations
    let experiment_diff = (experiments - positive_outcomes)
        .max(positive_outcomes);

    // Simplifying the factorial in the numerator of combinations
    let combinations_numer = (experiment_diff + 1
        ..=experiments)
        .map(BigUint::from)
        .product::<BigUint>();

    // Computing the remaining factorial in the denominator
    let combinations_denom = (1..=(experiments
        - experiment_diff))
        .map(BigUint::from)
        .product::<BigUint>();

    let negative_pow = experiments - positive_outcomes;

    // Multiplying combinations, p^k, q^n-k together
    let probability = Ratio::new_raw(
        combinations_numer
            * positive_numer.pow(positive_outcomes)
            * negative_numer.pow(negative_pow),
        combinations_denom
            * prob_denom.pow(positive_outcomes)
            * prob_denom.pow(negative_pow),
    );
```

```
pub fn moivre_laplace(
    experiments: u32,
    positive_outcomes: u32,
    positive_probability: FR,
    exponentiation_iterations: usize,
    square_root_iterations: usize,
) -> SolverResult {
    let now = Instant::now();

    let experiments: BigUint = experiments.into();
    let positive_outcomes: BigUint =
        positive_outcomes.into();
    let (positive_numer, prob_denom) =
        positive_probability.into();
    let negative_numer = &prob_denom - &positive_numer;

    let np = experiments * positive_numer;
    let two_npq_numer = 2u32 * &np * negative_numer;
    let two_npq_denom = &prob_denom * &prob_denom;

    // Since we need to subtract np from k, we also need to find the least common denominator and
    // scale numerators accordingly. Since denominator of k is equal to 1 we can just multiply it's
    // numerator by np's denominator and get the appropriate value
    let scaled_positive_outcomes =
        positive_outcomes * &prob_denom;

    // This if condition is here to prevent underflowing as we are working with unsigned numbers
    let exp_numer = if scaled_positive_outcomes >= np {
        scaled_positive_outcomes - np
    } else {
        np - scaled_positive_outcomes
    }
    .pow(2)
        * &two_npq_denom;

    let exp_denom = (&prob_denom).pow(2) * &two_npq_numer;

    let (exp_numer, exp_denom) = exp(
        exp_numer,
        exp_denom,
        exponentiation_iterations,
    );

    // These big u128 numbers are an approximation of pi in fraction form
    let root_numer =
        two_npq_numer * 30_246_273_033_735_921u128;
    let root_denom =
        two_npq_denom * 9_627_687_726_852_338u128;

    // function sqrt returns values in order numer, denom. But since we need 1 over sqrt, we just
    // swap around the numer and denom
    let (left_denom, left_numer) = sqrt(
        root_numer,
        root_denom,
        square_root_iterations,
    );

    let probability = Ratio::new_raw(
        left_numer * exp_denom,
        left_denom * exp_numer,
    );

    let elapsed = now.elapsed();

    SolverResult {
        took: Duration::microseconds(
            elapsed.as_micros().try_into().unwrap(),
        ),
        probability: GenericFraction::Rational(
            fraction::Sign::Plus,
            probability,
        ),
        iterations: 0,
    }
}
```

Відповідний інтерфейс:



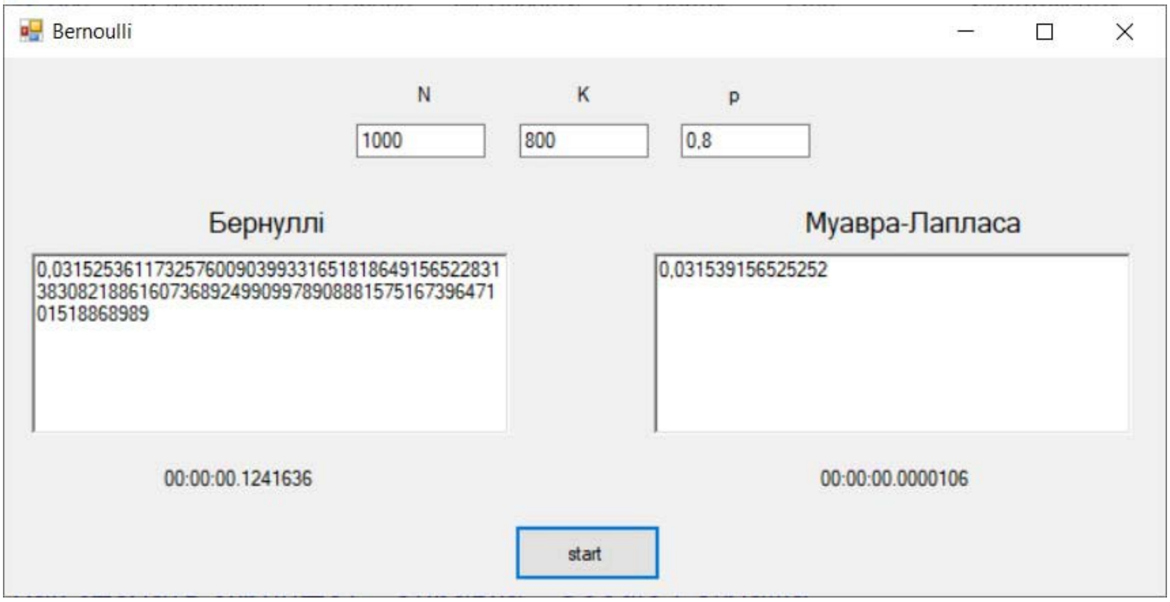
Джерело: розробка авторів.

Рис. 1

Формула Бернуллі та Муавра-Лапласа на C# та відповідний інтерфейс наведено нижче:

```
public void FBernuli()
{
    var sw = new Stopwatch();
    sw.Start();
    int N = Convert.ToInt32(textBox1.Text);
    BigInteger bigResultN = 1;
    int K = Convert.ToInt32(textBox2.Text);
    BigInteger bigResultK = 1;
    BigInteger bigResultNK = 1;
    String sp = textBox3.Text;
    double p = Convert.ToDouble(textBox3.Text);
    BigFloat QQQ = 1 - (BigFloat)p;
    int pp = sp.Length - 2;
    for (int i = 1; i <= N; i++)
    {
        bigResultN = bigResultN * i;
    }

    for (int i = 1; i <= K; i++)
    {
        bigResultK = bigResultK * i;
    }
    for (int i = 1; i <= N - K; i++)
    {
        bigResultNK = bigResultNK * i;
    }
    BigFloat BernuliResult = new BigFloat(0);
    BernuliResult = new BigFloat(bigResultN, bigResultK * bigResultNK);
    BernuliResult *= BigFloat.Pow((BigFloat)p, K);
    BernuliResult *= BigFloat.Pow((BigFloat)QQQ, (N - K));
    richTextBox1.Text = BernuliResult.ToString();
    sw.Stop();
    label3.Text = sw.Elapsed.ToString();
}
```

```
public void FLaplas(int n, int k, double p, double q)
{
    var sw = new Stopwatch();
    sw.Start();
    double I = 1 / Math.Sqrt(2 * Math.PI * n * p * (1 - p));
    double II = -(Math.Pow(k - n * p, 2) / (n * p * (1 - p))) / 2;
    double Lap = I * Math.Pow(Math.E, II);
    sw.Stop();
    label4.Text = sw.Elapsed.ToString();
    richTextBox2.Text = Lap.ToString();
}
```



Джерело: розробка авторів.

Рис. 2

Як видно з наведених результатів обчислень (рис. 1, рис. 2), підрахунок ймовірності того, що при n незалежних випробуваннях подія A відбудеться рівно k разів за формулою Бернуллі

$$P_n(k) = C_n^k p^k q^{n-k} \tag{1}$$

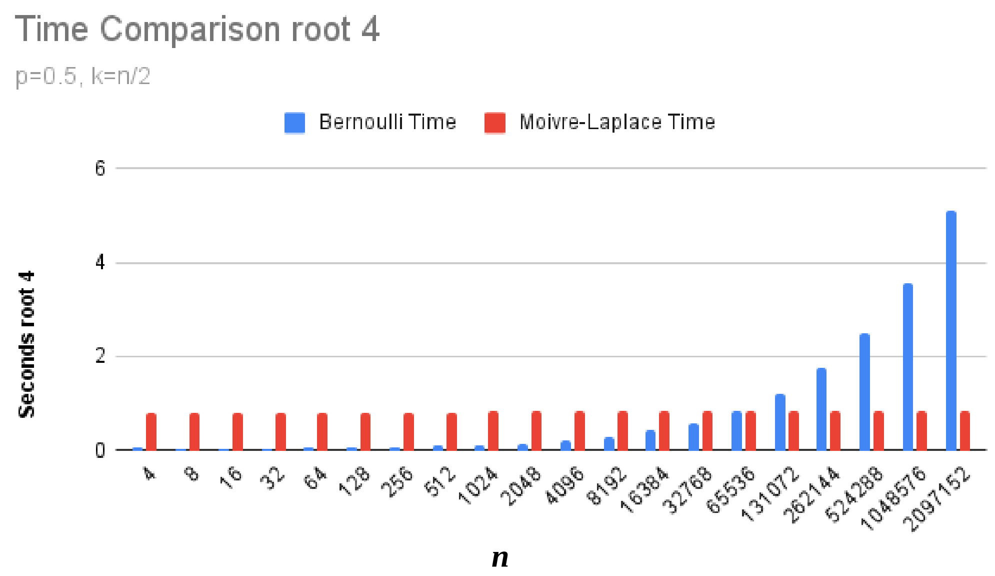
при n = 1000 забирає менше секунди часу в програмах написаних обома мовами.

Ця ж ймовірність знайдена за локальною теоремою Муавра-Лапласа

$$P_n(k) = \frac{1}{\sqrt{2\pi npq}} e^{-\frac{(k-np)^2}{2npq}} \tag{2}$$

потребує завжди менше секунди (рис. 3).

Щоб деталізувати час виконання, наведено графік для $\sqrt[4]{t}$, t – час виконання. Узагальнений графік часу виконання (рис. 3) показує, що до значення n = 65 536 час реалізації коду для формули Бернуллі (1) менший за час необхідний для локальної теореми Муавра-Лапласа (2) і лише після n = 524 288 починає зростати. Це підтверджує актуальність наближення формули Бернуллі (1) за допомогою локальної теореми Муавра-Лапласа (2).



Джерело: розробка авторів.

Рис. 3

Проте варто зауважити, що умови застосування локальної теореми Муавра-Лапласа в різних джерелах вказані з досить різним ступенем точності та зрозумілості. Наприклад, в [10] припускають, що числа n і k одночасно прямують до нескінченності так, що нормоване та центроване значення k є обмеженим:

$$x_{nk} \equiv (k - np)/\sqrt{npq} = O(1), n, k \to \infty. \quad (3)$$

Тоді $P_n(k)$ асимптотично еквівалентна $\frac{1}{\sqrt{2\pi npq}} \exp\left(-\frac{x_{nk}^2}{2}\right)$ (тобто їх відношення прямує до 1) при $n, k \to \infty$.

В [2] умовою твердження є

$$m - np = o\left(n^{\frac{2}{3}}\right), \quad (4)$$

а в [11]:

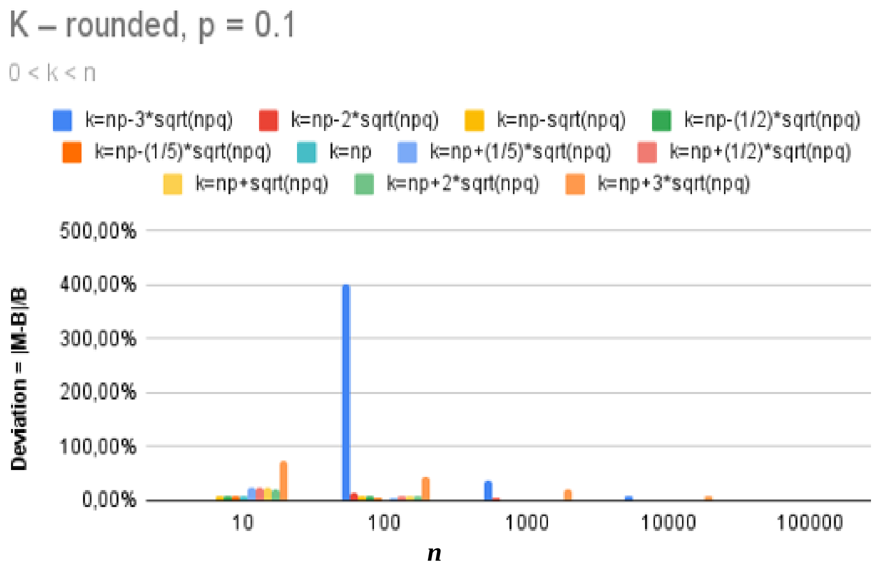
$$|m - np| = o(npq)^{\frac{2}{3}}. \quad (5)$$

Умови (3)–(5) не очевидні для практичного користувача на відміну від наведеної у Вікіпедії [12]: *якщо $n \to \infty$, то для k* з $\sqrt{npq}$*-околу точки np, буде справджуватися дане наближення*. Але вже в наступному реченні наводиться наступне «уточнення» [12]: «практичний зміст теореми Муавра-Лапласа простий: при великих n ймовірність можна розраховувати за формулою:

$$P_n(k) = \frac{1}{\sqrt{2\pi npq}} e^{-\frac{(k-np)^2}{2npq}}. \quad (4)$$

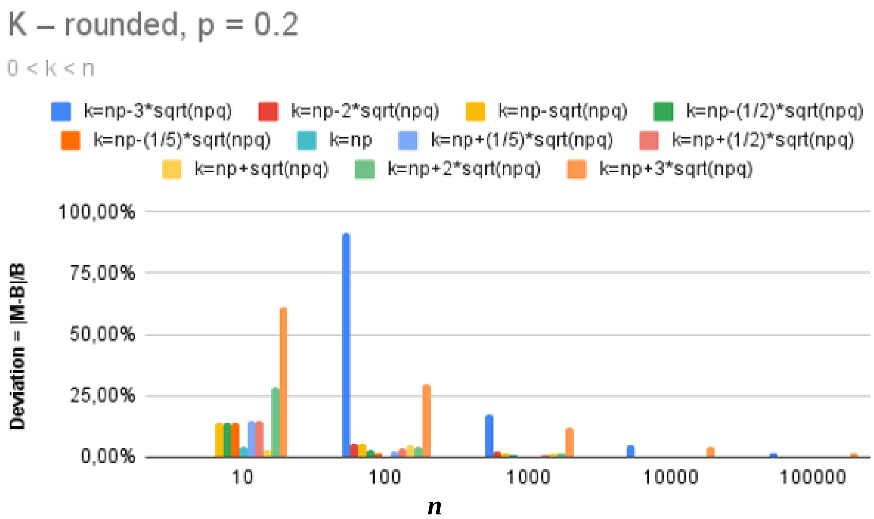
В [9] стверджується, що «формула (4) дає добре наближення, якщо n достатньо велике, p і q не дуже близькі до нуля, $npq > 9$». Замість 9 зустрічаються і інші числа. Подивимось як виглядають відносні похибки обчислень ймовірностей за (1) і (2) при $n = 10, 100, 1000, 100000$; $p = 0.1, 0.2, 0.3, 0.4, 0.5$ та

$$k = np;\ np \pm \frac{1}{5}\sqrt{npq};\ np \pm \frac{1}{2}\sqrt{npq};\ np \pm \sqrt{npq};\ np \pm 2\sqrt{npq};\ np \pm 3\sqrt{npq}.$$



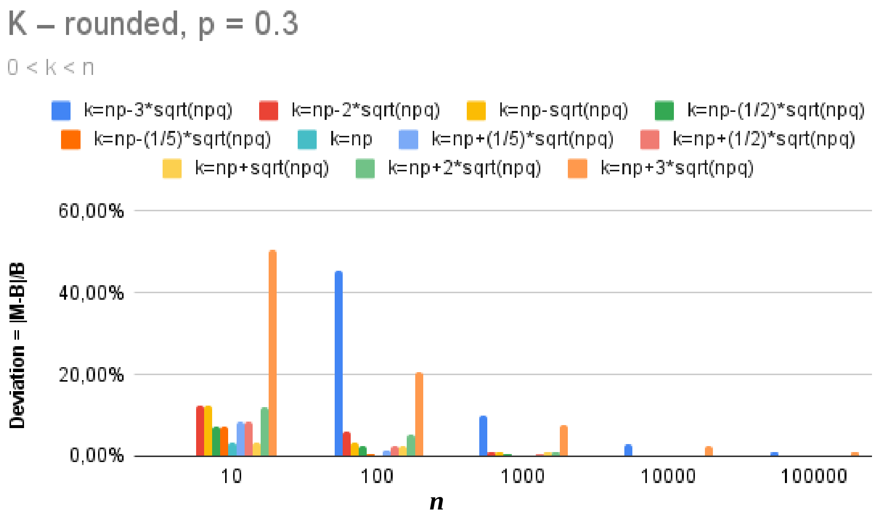
Джерело: розробка авторів.

Рис. 4



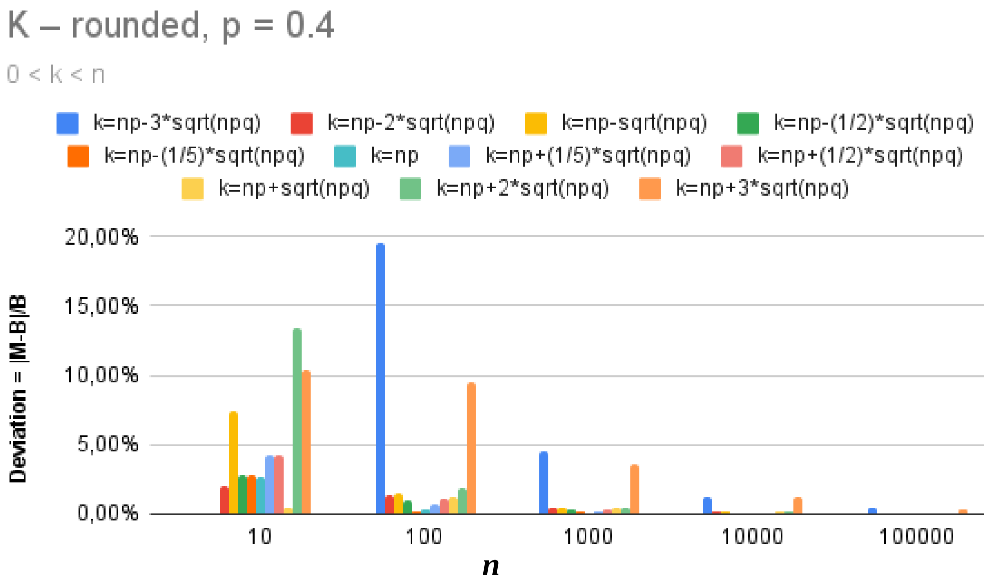
Джерело: розробка авторів.

Рис. 5



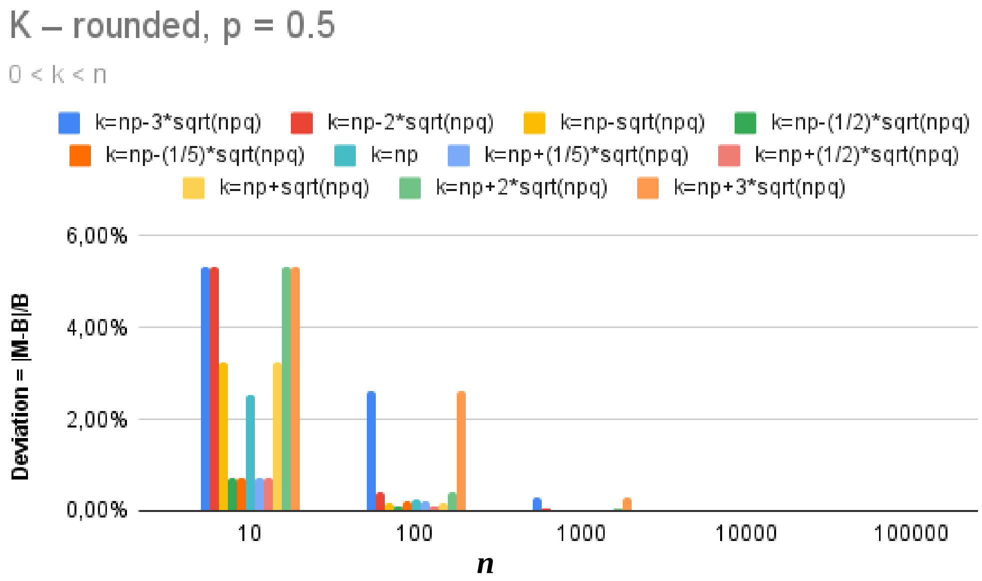
Джерело: розробка авторів.

Рис. 6



Джерело: розробка авторів.

Рис. 7



Джерело: розробка авторів.

Рис. 8

Наведені графіки (рис. 4–8) підтверджують, що при малих $n = 10$ локальна теорема Муавра-Лапласа дає не дуже достовірний результат. При $n > 100$ результати стають достовірними з відносною похибкою 5%, якщо $k \in \left(np - \sqrt{npq};\ np + \sqrt{npq}\right)$. А при $n > 10000$ інтервал, на якому результати будуть достовірними може бути розширений до $\left(np - 3\sqrt{npq};\ np + 3\sqrt{npq}\right)$. При цьому, якщо накладати умову виключно на npq ($npq > 9$, [9]), твердження не є правильним. Наприклад, при $n = 100, p = 0.2$, відносна похибка складає близько 90%, при $n = 1000, p = 0.2$ – майже 20% (рис. 5). Цікаво також, що при p близькому до нуля (у нас 0.1) відносна похибка при npq =16 дорівнює рекордним 400% (рис. 4).

Висновки. Застосовуючи локальну теорему Муавра-Лапласа слід пам'ятати, що її використання обмежене тими k, що належать $\sqrt{npq}$ -околу точки np. Чим далі знаходиться k від вказаного околу, тим більшу похибку має результат. Підтверджено, що умови на кшталт $npq > 9$ без вказівки околу для k теж дають хибні результати. Швидкість обчислень сучасними комп'ютерами така висока, що лише після n = 65 536 час для підрахунку ймовірності за формулою Бернуллі стає більшим, що значно змінює уявлення про «великі n». Здійснені етапи роботи корисні набутими навичками в програмуванні та в проведенні експериментальних досліджень. Отримані результати можуть в подальшому використовуватися викладачами та студентами для навчальних цілей, а також іншими для практичних розрахунків [13].

Список використаної літератури

1. Kelbert M. Ya., Sukhov Yu. M. Probability and Statistics by example. Vol. I: Basic probability and statistics. Cambridge University Press, 2005. 373 p.
2. Гнеденко Б. В. Курс теорії ймовірностей: підручник. К.: ВПЦ "Київський університет", 2010. 464 с.
3. Klenke A. Probability Theory. A comprehensive course. Springer Cham, 2020. 716 p.
4. Stirzaker D. Elementary Probability. Cambridge University Press, 2003. 524 p.
5. Клесов О. І. Вибрані питання теорії ймовірностей та математичної статистики: навч. посіб. Київ: "ТВіМС", 2010. 248 с.
6. Барковський В. В. Теорія ймовірностей та математична статистика. К.: ЦНЛ, 2019. 448 с.
7. Жалдак М. І., Кузьміна Н. М., Михалін Г. О. Теорія ймовірностей і математична статистика: підручник для студентів фізикоматематичних спеціальностей педагогічних університетів. Вид. 2, перероб. і доп. Полтава: "Довкілля-К", 2009. 500 с.
8. Тичинська Л. М., Черепащук А. А. Теорія ймовірностей: навчальний посібник. Вінниця: ВНТУ, 2010. 112 с.
9. Турчин В. М. Теорія ймовірностей і математична статистика. Основні поняття, приклади, задачі: підручник для студентів ВНЗ. Дніпропетровськ: ІМА-прес, 2014. 556 с.
10. Карташов М. В. Імовірність, процеси, статистика: навч. посіб. К.: Видавничо-поліграфічний центр "Київський університет", 2008. 494 с.
11. Голіченко І. І., Ільєнко М. К., Савич І. М. Вступ до теорії ймовірностей: підручник для здобувачів ступеня бакалавра за спеціальністю 111 Математика. Київ: КПІ ім. Ігоря Сікорського, 2022. 221 с.
12. Локальна теорема Муавра-Лапласа. *Вікіпедія*. URL: wikipedia.org/Локальна теорема Муавра-Лапласа.
13. Бобер Д. Сайт для обчислення ймовірностей за формулою Бернуллі та за локальною теоремою Муавра-Лапласа. Bernoulli vs Moivre Laplace. bernoulli-vs-moivre-laplace.netlify.app.