МІНІСТЕРСВО ОСВІТИ І НАУКИ УКРАІНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп'ютерних наук та кібернетики

ЛАБОРАТОРНА РОБОТА №3

з навчальної дисципліни

«*Чисельні методи*»

на тему:

*«РОЗВ’ЯЗУВАННЯ СИСТЕМ ЛІНІЙНИХ АРИФМЕТИЧНИХ РІВНЯНЬ»*

*варіант 8*

Виконав:

студент ІІ курсу, групи К-27

спеціальності «Комп’ютерні науки. Інформатика»

*Некряч Владислав*

*Київ, 2021 рік*

ЗМІСТ

[1. Метод Якобі 3](#_Toc68032012)

[2. Метод Зейделя 8](#_Toc68032013)

1. Метод Якобі

**Теоретичні відомості**

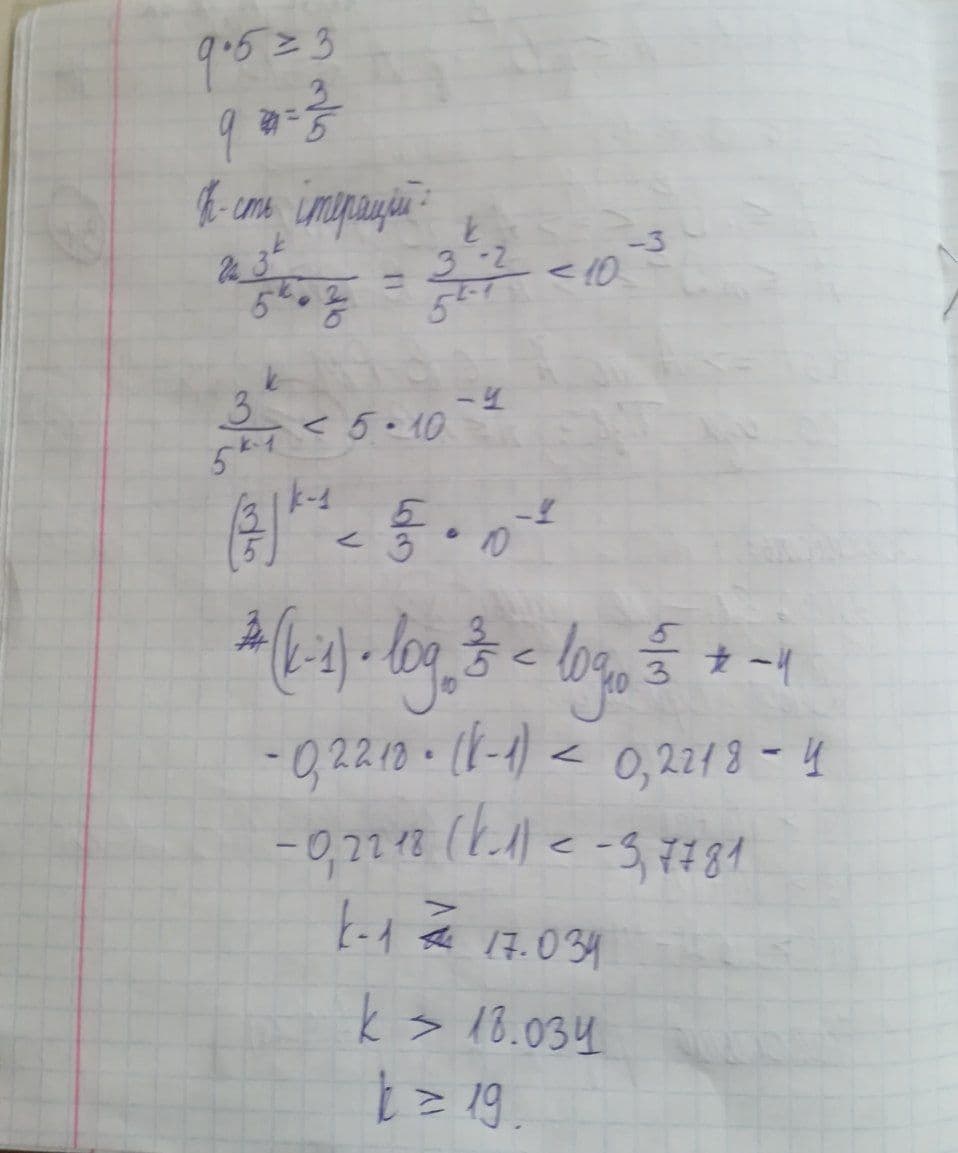
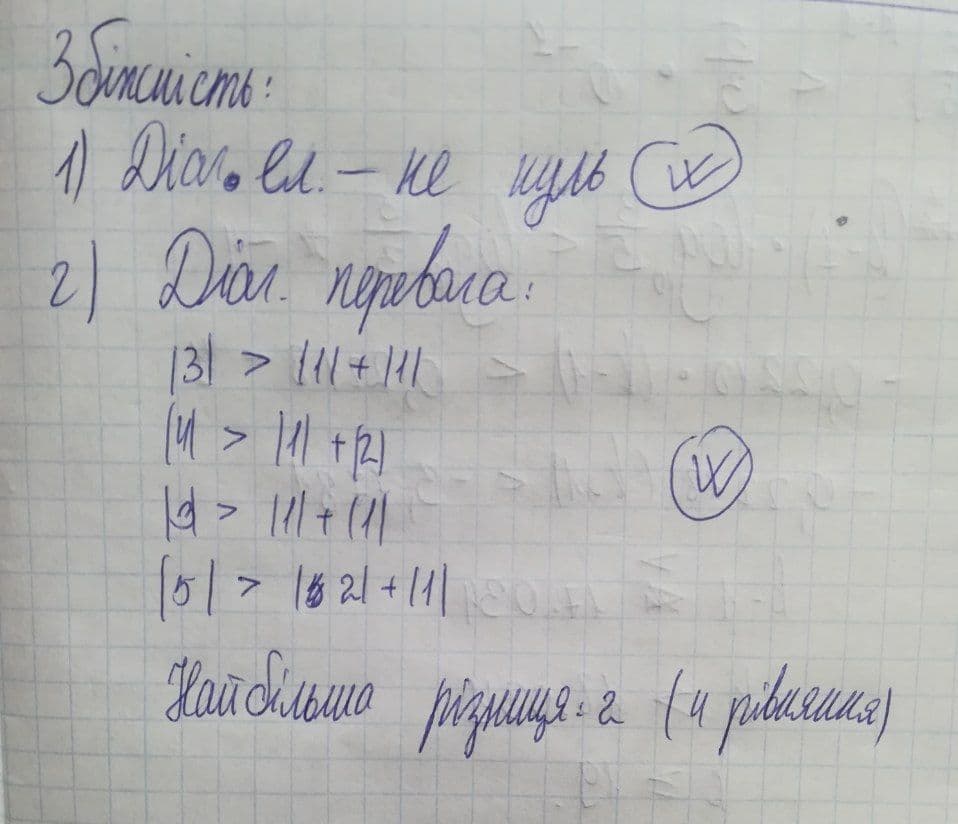
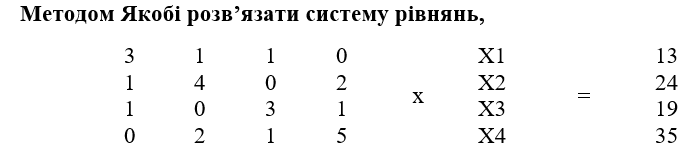
Припустимо, що діагональні коефіцієнти невиродженої матриці ненульові (). Якщо деякі , то цього можна досягти, переставивши деякі рядки чи стовпці матриці. Розділивши -те рівняння на , то отримаємо таку СЛАР:

Задамо якесь початкове наближення . Наступні наближення обчислимо за формулами

Метод збігається, тобто , якщо виконуються умови діагональної переваги матриці . Якщо ж виконуються нерівності , то правдива така оцінка точності:

Ітерації виконують, поки не буде отримано потрібну кількість цифр у компонентах розв’язку чи до виконання умови .

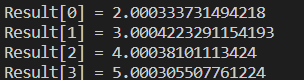
Вибір останньої умови пояснюється тим, що в разі її виконання для маємо оцінку

**Постанова задачі**

Для даної задачі умова виконується при k >= 19.

Дана задача – збіжна.

**Розв’язок**

****

**Код:**

"use strict";

let dimensions = 4;

let matrix = [

  [3, 1, 1, 0],

  [1, 4, 0, 2],

  [1, 0, 3, 1],

  [0, 2, 1, 5]];

let b = [13, 24, 19, 35];

let result = [0, 0, 0, 0];

let epsilon = 0.001;

let currentVector = [0, 0, 0, 0];

for (let q = 0; q <= 20; q++) {

  for (let i = 0; i < dimensions; i++) {

    // fixate element from solution vector

    currentVector[i] = b[i];

    // do subtractions

    for (let j = 0; j < dimensions; j++) {

      if (i != j)

        currentVector[i] -= matrix[i][j] \* result[j];

    }

    // divide by major element

    currentVector[i] /= matrix[i][i]

  }

for (let i = 0; i < dimensions; i++) {

    // push updated element back to result

    result[i] = currentVector[i]

  }

}

for (let i = 0; i < dimensions; i++){

  console.log(`Result[${i}] = ${currentVector[i]}`);

}

1. Метод Зейделя

**Теоретичні відомості**

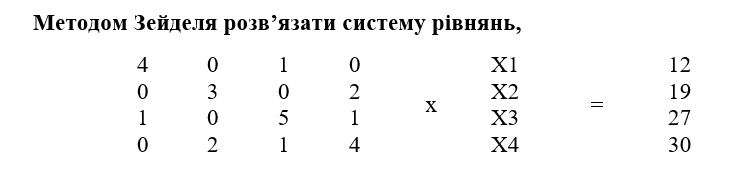
Якщо в сумі

Використати вже відомі нові значення , то отримаємо формулу

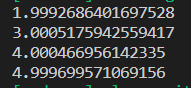
Достатні умови збіжності методу Зейделя такі самі, як методу Якобі. Крім того метод Зейделя збігається, якщо . Умова невід’ємності симетричної матриці означає, що невід’ємні її головні мінори.

Змінивши порядок обчислення компонент, отримаємо ще одну формулу методу Зейделя:

**Постанова задачі**



**Розв’язок**



"use strict";

let dimensions = 4;

let matrix = [

  [3, 1, 1, 0],

  [1, 4, 0, 2],

  [1, 0, 3, 1],

  [0, 2, 1, 5]];

let b = [13, 24, 19, 35];

let result = [0, 0, 0, 0];

let subtractionSum = 0;

let epsilon = 0.001;

let iterationDelta = 0

do {

  iterationDelta = 0;

  for (let i = 0; i < dimensions; i++) {

    const currentElement = result[i];

    subtractionSum = 0;

    // basic solving technique of lin. eq.

    for (let j = 0; j < dimensions; j++) {

      if (j != i)

        subtractionSum += matrix[i][j] \* result[j];

    }

    result[i] = (b[i] - subtractionSum) / matrix[i][i];

    // find biggest delta

    let currentDelta = Math.abs(result[i] - currentElement)

    if (currentDelta > iterationDelta)

      iterationDelta = currentDelta;

  }

} while (iterationDelta > epsilon);

for (let i = 0; i < dimensions; i++) {

  console.log(`${result[i]}`);

}

//# sourceMappingURL=index.js.map