НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет прикладної математики
Кафедра прикладної математики

Звіт
із лабораторної роботи №4
із дисципліни «Розподілені і хмарні обчислення»

Виконав:

студент групи КМ-01

Романецький М.С.

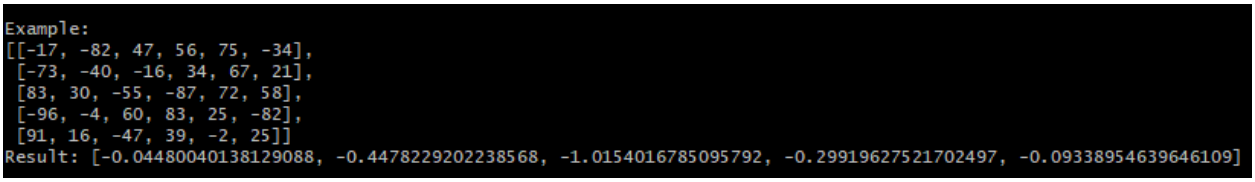
Керівник:

Доцент кафедри ПМА
Ліскін В. О.

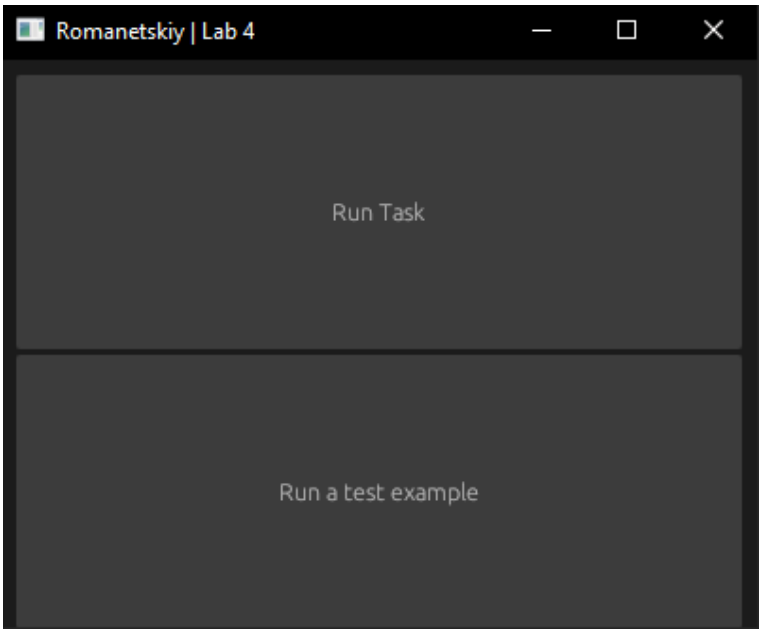
Київ — 2023

Мета роботи: Розпаралелити метод Гауса для вирішення СЛАР
Опис програми: Бібліотека ‘Rayon’ буде використовуватись для паралелізму. Для порівняння часу роботи буде використовувати ‘Criterion’

Розміри матриць	Потоки	Середній час виконання
10	1	16.565 мкс
10	2	30.497 мкс
10	4	18.779 мкс
10	8	25.194 мкс
100	1	2.0161 мс
100	2	2.3203 мс
100	4	1.0822 мс
100	8	1.1092 мс
500	1	868.46 мс
500	2	701.80 мс
500	4	711.77 мс
500	8	655.97 мс
1000	1	6.5014 с
1000	2	5.7763 с
1000	4	5.6477 с
1000	8	5.5008 с



Тестовий приклад



Інтерфейс
Висновки: Розпаралелення ефективно лише на великих об’ємах даних

Код програми:

```
// cd D:/KPI/Distributed_computing/Labs/lab_4

use lab_4::lineareq::gauss;
use ndarray::prelude::*;
use eframe::{epi, egui::{self, CtxRef}};
use std::process::Command;
use egui::vec2;


struct MyApp {
    //  дані та стан програми
}

impl Default for MyApp {
    fn default() -> Self {
        Self {
            // Ініціалізація стану
        }
    }
}

impl epi::App for MyApp {
    fn name(&self) -> &str {
        "Romanetskiy | Lab 4"
    }

    fn update(&mut self, ctx: &CtxRef, _frame: &mut epi::Frame) {
        egui::CentralPanel::default().show(ctx, |ui| {
            let button_size = vec2(ui.available_width(), 145.0);
            if ui.add_sized(button_size, egui::Button::new("Run Task")).clicked()
{
                task()
            }
            if ui.add_sized(button_size, egui::Button::new("Run a test
example")).clicked() {
                example()
            }
        });
    }
}


fn example() {
    let m = array![
        [-17., -82., 47., 56., 75., -34.],
        [-73., -40., -16., 34., 67., 21.],
        [83., 30., -55., -87., 72., 58.],
        [-96., -4., 60., 83., 25., -82.],
        [91., 16., -47., 39., -2., 25.],
    ];
```

```
    println!("\nExample: \n{}", m);

    let m = gauss(m);
    println!("Result: {}", m);
}


fn task() {
    println!("\nTask:");
    println!("Запускаємо cargo bench");
    match Command::new("cargo").args(&["bench"]).status() {
        Ok(status) => if status.success() {
            println!("cargo bench виконано успішно");
        } else {
            eprintln!("cargo bench завершилося з помилкою");
        },
        Err(e) => eprintln!("Помилка при запуску cargo bench: {}", e),
    }
}


fn main() {
    let app = MyApp::default();
    let mut native_options = eframe::NativeOptions::default();
    native_options.initial_window_size = Some(egui::vec2(400.0, 300.0));
    eframe::run_native(Box::new(app), native_options);
}
```