Barzowska Monika, 238143
Bienias Jan, 238201
3I TE1

16.12.2017

Algorytmy numeryczne

Zadanie 3: Grzybobranie

1. Wstęp

Celem zadania była implementacja zadanej gry w grzybobranie, która wymagała przeprowadzenia operacji na macierzach przy użyciu:
- języka `C++` i biblioteki `Eigen3` wspierającej operacje na macierzach,
- implementacji niezbędnych funkcji w wybranym języku programowania (tu `C#`).

2. Materiały i metody

Testy zostały przeprowadzone dla dwóch rodzajów danych wejściowych - z kostką symetryczną, o równomiernym rozkładzie prawdopodobieństwa oraz dowolną.
Obliczenia generowane w `C#` zostaly zestawione z wynikami generowanymi w `C++ Eigen`.

3. Wyniki i wnioski

Wyniki obliczeń zostały uśrednione z 5 próbek na każdy rozmiar macierzy. Taka ilość wystarczała, aby zauważyć tendencje zmian.

3.1. Implementacja metod iteracyjnych

Rozmiar macierzy	Wynik Monte Carlo	Wynik Gaussa partial bez optymalizacji	Wynik Gaussa partial z optymalizacją	Wynik Jacobi	Wynik Gauss-Seidel
1624	0,727665	0,727559231817524	0,727559231817524	0,727559231817522	0,727559231817521
3272	0,626816	0,626955397543633	0,626955397543633	0,626955397543631	0,626955397543632

Tabela 1: Zestawienie wyników poszczególnych metod, w tym iteracyjnych, służących do rozwiązywania układów równań liniowych (wykorzystano dane dotycząe przebiegu gry przy kostce symentrycznej).

Wyniki metod porównano z wynikami Monte Carlo; wynik tego porównania implikuje poprawność implementacji metod iteracyjnych. Wynik metody Monte Carlo był obliczany każdorazowo dla miliona próbek.

3.2. Optymalna ilość iteracji, użycie normy wektora

W celu zmienjszenia czasu potrzebnego do uzyskania wyniku metodami iteracyjnymi warto wykorzystać algorytm, który liczy średnią różnicę sum wektora otrzymanego w poprzedniej oraz obecnej iteracji, następnie sprawdza, czy różnica ta jest mniejsza od zadanego ε i jeśli jest, nie pozwala na dalsze iterowanie. [1]
Pozwala nam to jednoznacznie ustalić wymaganą ilość iteracji oraz iterować do jej osiągnięcia, co pozwoli nam uniknąć dalszych, zbędnych obliczeń.

3.3. Optymalny dobór metody

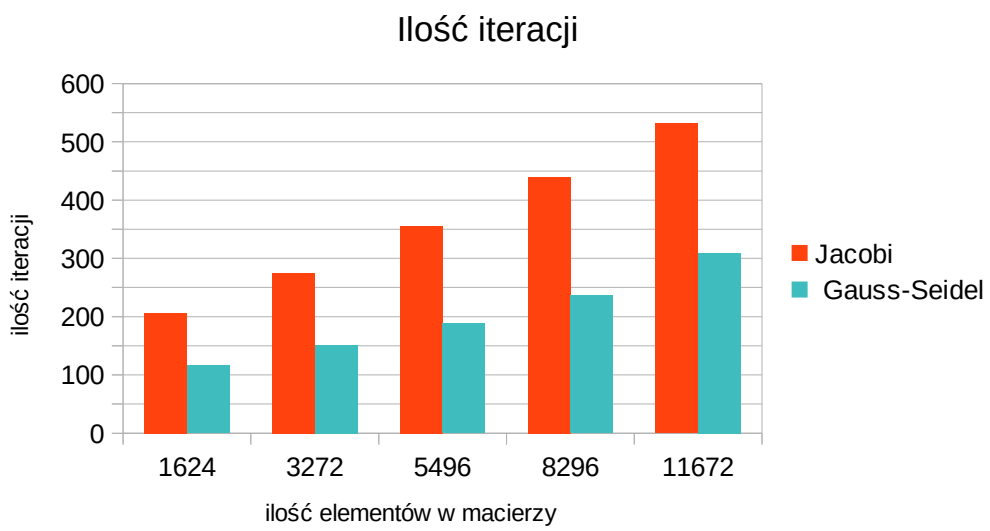
Porównanie wyników i czasu działania następujących metod:
-metoda Gaussa z częściowym wyborem elementu podstawowego,
-metoda Gaussa z częściowym wyborem elementu podstawowego z optymalizacją dla macierzy rzadkich,
-metoda iteracyjna Gaussa-Seidela,
-metoda iteracyjna Jacobiego,
-metoda z biblioteki Eigen3: partialPivLu, z częściowym wyborem elemntu podstawowego,
-metoda z biblioteki Eigen3: SparseLU, z częściowym wyborem elementu podstawowego z optymalizacją dla macierzy rzadkich.

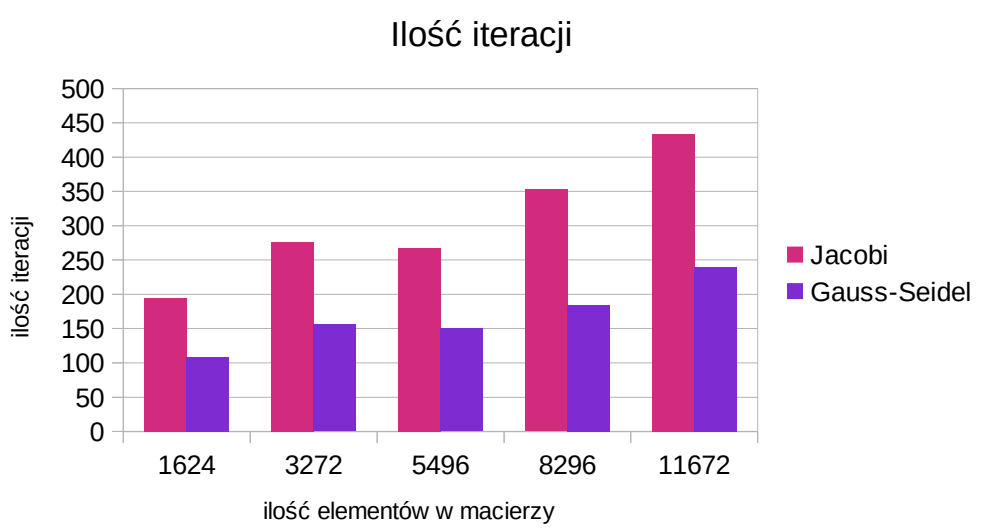
3.3.1. Testy przeprowadzone dla kostki symetrycznej

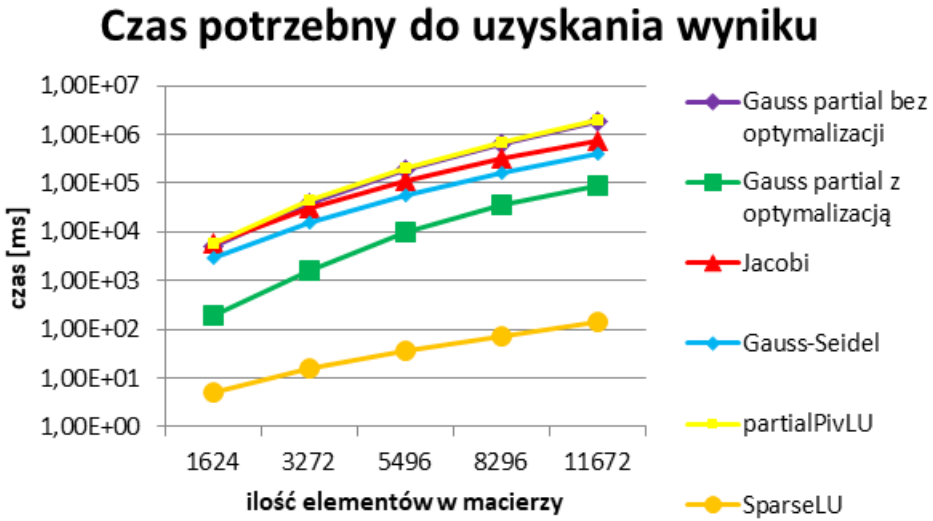
Rozmiar planszy (minimalny - zadany)	Ilość grzybów	Wartości na kostce
7	4	-4 ,-3, -2, -1, 0, 1, 2, 3, 4
Pozycje graczy	Miejsca występowania grzybów	Prawdopodobieństwo wartości kostki
3, 4	1, 2, 5, 6	1, 1, 1, 1, 1, 1, 1, 1, 1

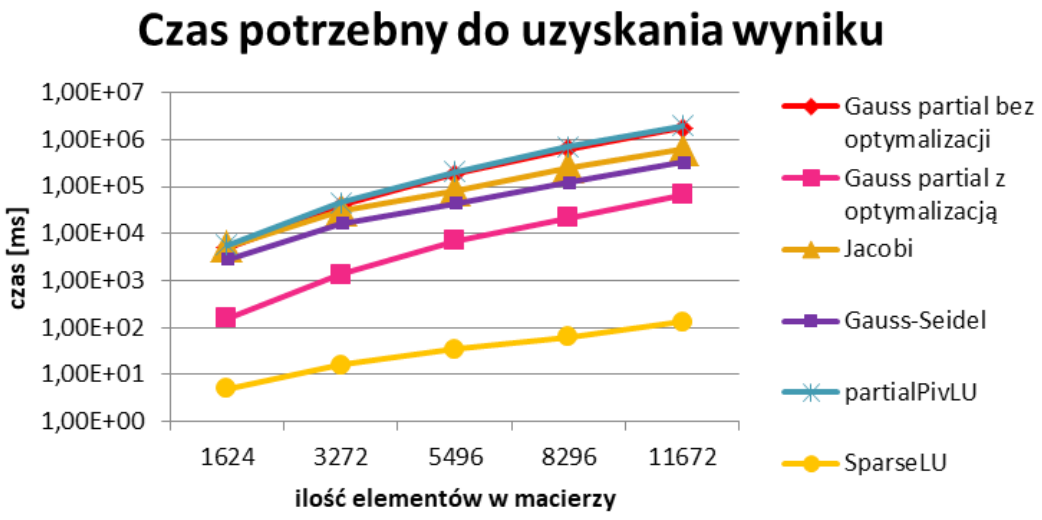
3.3.2. Testy przeprowadzone dla kostki dowolnej

Rozmiar planszy (minimalny - zadany)	Ilość grzybów	Wartości na kostce
7	4	3, 4, 5, 2, 7, 9, 1, 8, 6
Pozycje graczy	Miejsca występowania grzybów	Prawdopodobieństwo wartości kostki
3, 4	1, 2, 5, 6	8, 3, 1, 23, 4, 5, 6, 7, 69









3.4. Wnioski

Powższe wykresy prezentują ilości iteracji oraz czasy potrzebne do uzyskania wyniku daną metodą. Rozmiar macierzy zależny jest od liości wygenerowanych stanów, co zależy od rozmiaru mapy (7-15 pól) oraz od ilości grzybów (stała, 4).

A. Ilość iteracji

Dla danego typu kostki metoda iteracyjna Gaussa-Siedela potrzebuje około 2 razy mniej iteracji (a co za tym idzie – około dwa razy mniej czasu) w porównaniu z metodą iteracyjną Jacobiego.

Dodatkowo, jak można jednak zauważyć, rozpatrując kostkę dowolną, ilość iteracji potrzebna dla macierzy o danym rozmiarze jest niższa niż ilość iteracji potrzebna do obsługi macierzy tej samej wielkości powstałej przez zadanie kostki symetrycznej.

B. Czas

Funkcja z biblioteki `Eigen3, SparseLU` charakteryzuje się najkrótyszym czasem działania. Dzieję się tak m.in. dlatego, że jako jedną za składowych dostaje wektor z informacją o ilościach niezerowych wartości w kolumnach macierzy.

C. Dokładność wyniku

We wszystkich metodach wyniki pokrywały się prawie idealnie – błędy pojawiały się na 15 miejscu po przecinku; warto zauważyć, że w metodach iteracyjnych miało to związek z zadanym $\varepsilon = 1{,}00\text{E-}15$. Wysoka prezycja wyniku jest związana z bardzo małą iloscią wartości niezerowych w macierzach, co zmniejsza ryzyko błędu w precyzji obliczeń.

4. Szczegóły wykonania zadania

Podział obowiązków	
Barzowska Monika	**Bienias Jan**
Pobieranie z `C#` i generowanie wyników w `Eigen3`	Implementacja Monte Carlo
Implementacja metod iteracyjnych	Optymalizacja metod iteracyjnych i Gaussa
Ustalenie algorytmu generacji stanów gry	
Generacja stanów gry	
Testy i generowanie wyników	Generacja układu równań
Obróbka wyników, wykresy i opracowanie	Napisanie skryptów do generowania wyników

5. Źródła

[1] http://web.cecs.pdx.edu/~gerry/class/ME448/notes_2012/pdf/stoppingCriteria.pdf