Algorytmy numeryczne

Zadanie 4
Dawid Bińkuś & Oskar Bir & Mateusz Małecki
grupa 1 tester-programista

13 Styczeń 2019

1 Aproksymacja

Sprawozdanie prezentuje analizę aproksymacji dla problemu określonego w zadaniu 3. W tym celu, zastosowana została aproksymacja dla metod testowanych w zadaniu 3:

- Metoda Gaussa (PG) - wielomian 3-go stopnia,
- Metoda Gaussa z drobną optymalizacją dla macierzy rzadkich (SPG) - wielomian 2-go stopnia,
- Metoda Gaussa-Seidela (GS) przy założonej dokładności 1e-10 - wielomian 2-go stopnia,

  Oraz dodatkowo:

  – Metoda zaimplementowana w oparciu o macierze rzadkie (S) - wielomian 1 stopnia (wykonane za pomocą LUDecomposition z biblioteki Apache Commons Math[1]) - wariant z użyciem własnego typu danych (SparseFieldMatrix),
  – wariant z użyciem tablicy double (DS) (OpenMapRealMatrix)

Program na potrzeby analizy problemu został napisany w języku Java. Ilość agentów oznaczana jest jako N

2 Próbka pomiarów czasu

2.1 Zakres testów

Na potrzeby wyliczenia funkcji aproksymacyjnej dla każdej z metod, wykonane zostały testy dla $N = 15, 16, ..., 60$, co przy $N = 60$ odpowiada liczbie około 2000 równań. Dla mniejszej ilości agentów testy wydajnościowe zostały wykonane kilka razy dla uśredniania wyniku.

2.2 Wyniki

Wyniki zostały zamieszczone w pliku csv

3 Aproksymacja średniokwadratowa dyskretna

Za pomocą aproksymacji średniokwadratowej dyskretnej, wygenerowane zostały wielomiany dla każdego typu pomiarów. Prezentują się one w sposób następujący:

1. Generowanie macierzy:
   (a) $(2.3350439175615983\text{e}{-11})x^3 - (3.73562735299031\text{e}{-8})x^2 + (3.927924861861045\text{e}{-5})x^1 - 0.006632202623758287$ dla PG,
   (b) $0.026544550834587136x^1 - 12.310426442549858$ dla S (wariant dla wielomianu stopnia 1, $m = 1$),
   (c) $(1.1160411337156506\text{e}{-8})x^3 - (1.138102839915528\text{e}{-5})x^2 + 0.006355108980340516x^1 - 0.9349542064725634$ dla S (wariant dla wielomianu stopnia 3, $m = 3$),
   (d) $0.005862735639798491x^1 - 2.8318326460105085$ dla S (wariant)
   (e) $(3.905262387830365\text{e}{-8})x^2 - (2.5264926661242525\text{e}{-5})x^1 + 0.006283510185750934x^0$ dla SPG,
   (f) $(5.486623477461811\text{e}{-8})x^2 - (5.4389374355751846\text{e}{-5})x^1 + 0.014501100847044942$ dla GS.
2. Rozwiązywanie układu równań:

[1] `http://commons.apache.org/proper/commons-math/javadocs/api-3.6/overview-summary.html`

(a) $(2.1592223868228134e{-}8)x^3 - (3.716638790505861e{-}5)x^2 + 0.023672650455756852x^1 - 3.8108135102259824$ dla PG,

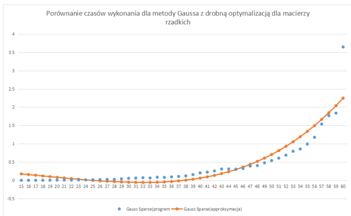
(b) $(1.583603905695412e{-}4)x^1 - 0.06429601760699291$ dla S,

(c) $(2.0353485741938283e{-}5)x^1, -0.008158601334700392$ dla DS,

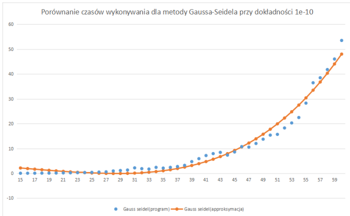
(d) $(1.3001654605220337e{-}6)x^2 - 0.0014560130726117273x^1 + 0.3507906257127139$ dla SPG,

(e) $(2.3048124134271384e{-}5)x^2 - 0.020571995359005252x^1 + 4.514786569835568$ dla GS.

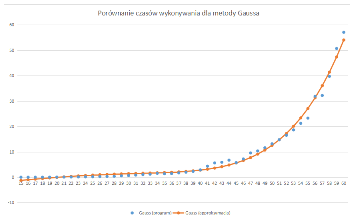
Rysunek 1: Wykresy reprezentujące czas wykonania i błędy bezwzględne zaimplementowanych algorytmów



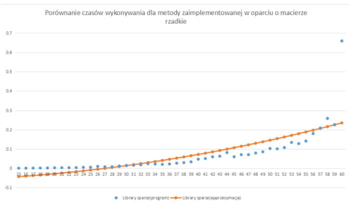
(a)



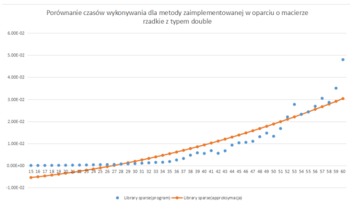
(b)



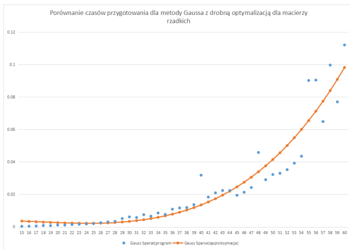
(c)



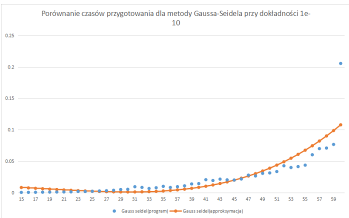
(d)



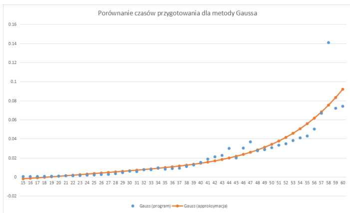
(e)



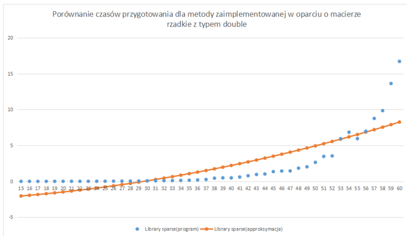
(f)



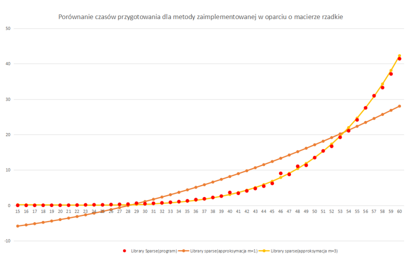
(g)



(h)



(i)



(j)

3.1 Poprawność uzyskanego rozwiązania

Błąd aproksymacji		
Metoda	Wariant	Błąd aproksymacji[s]
PG	Obliczanie	87.52296395468811
	Generowanie	0.005637166746569134
SPG	Obliczanie	2.932007861300984
	Generowanie	0.0035357258834377552
GS	Obliczanie	204.23823436655252
	Generowanie	0.013028009281415509
S	Obliczanie	0.2257609275320777
	Generowanie $m = 1$	3727.8124913151905
	Generowanie $m = 3$	26.142413480394573
DS	Obliczanie	8.484328415617502e−4
	Generowanie	208.62045033311406

Jak widać na załączonych wykresach 1 oraz tabeli powyżej prezentującej błąd aproksymacji, w większości przypadków funkcja aproksymująca poprawnie wylicza kolejne czasy wykonywania algorytmu.

Wyjątkiem jest funkcja dla generowania w metodzie S. Wielomian stopnia pierwszego okazał się być niewystarczający dla tego typu problemu, gdyż w bibliotece Commons Math maceirze rzadkie zapisywane są w postaci obiektów HashMap - co znacząco wydłuża czas

4 Ekstrapolacja

5 Próba obliczenia

6 Podział pracy

Dawid Bińkuś	**Oskar Bir**	**Mateusz Małecki**
Praca nad strukturą projektu.	Analiza algorytmu Gaussa oraz implementacja wariantu G	Implementacja typu własnej precyzji
Przygotowanie sprawozdania	Przygotowanie testów i ich uruchomienie	Operacje na macierzach
Implementacja algorytmu Gaussa w wariantach PG i FG	Analiza danych oraz określenie czasu pracy typu Fraction	Praca nad strukturą projektu
Implementacja generycznej klasy MyMatrix	Przygotowanie wykresów końcowych	Implementacja generycznej klasy MyMatrix