1. Jak uruchomić

Aby skompilować program użyć polecenia

make

Aby uruchomić rozwiązywanie układu równań metodą Jacobiego

make runj

Aby uruchomić rozwiązywanie układu równań metodą Gaussa-Seidela

make rung

Aby wyświetlić wykresy przedstawiające zbieżność metod

gnuplot

load “j.plt”

2. Sprawozdanie

Metody iteracyjne to metody numeryczne, które polegają na generowaniu ciągu liczb, które są coraz bliższe rozwiązaniu danego problemu matematycznego. Ciąg ten jest generowany przy użyciu powtarzających się operacji, aż do osiągnięcia zadanej dokładności. Jedynymi z nich są metody Jacobiego i Gaussa-Seidela służące do rozwiązywanie układu równań liniowych.

Metoda Jacobiego polega na użyciu poprzedniego przybliżenia rozwiązania, aby obliczyć nowe przybliżenie. W metodzie tej, rozwiązanie jest aktualizowane przez równoległe przetwarzanie każdej niewiadomej. Oznacza to, że dla każdej niewiadomej x_i, aktualizacja jest wykonywana na podstawie pozostałych nieznanych w poprzednim przybliżeniu.

Natomiast Metoda Gaussa-Seidela polega na użyciu bieżącego przybliżenia rozwiązania, aby obliczyć nowe przybliżenie. W metodzie tej, rozwiązanie jest aktualizowane przez sekwencyjne przetwarzanie każdej niewiadomej. Oznacza to, że dla każdej niewiadomej x_i, aktualizacja jest wykonywana na podstawie bieżącego przybliżenia dla pozostałych nieznanych.

Stąd, Metoda Gaussa-Seidela jest szybsza niż metoda Jacobiego, ponieważ jest ona w stanie wykorzystać nowe informacje dotyczące rozwiązania już po kilku iteracjach.

3. Wyniki

Metoda Gaussa-Seidela przy każdym nowym uruchomieniu (Za każdym razem nowe losowe wektory startowe) dochodzi do końca już w zaledwie ok. 40 iteracjach. W przypadku metody Jacobiego już po 170 iteracjach na wykresie (na samym dole) można zaobserwować szum, wynikający z powstałych błędów numerycznych. Widać że metoda Gaussa-Seidela zbiega o wiele szybciej.

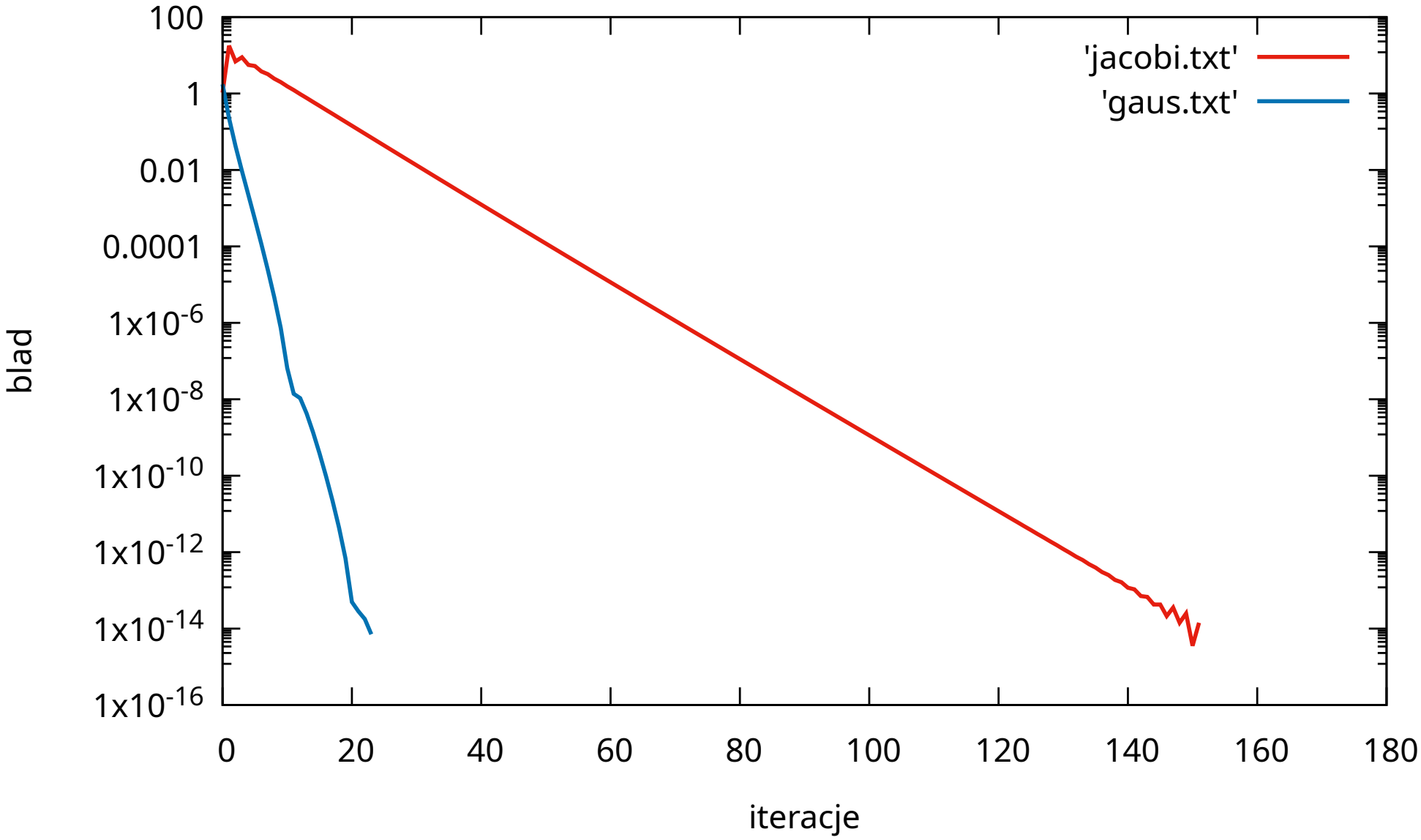
Wyniki programu (output konsoli) są następujące

Gauss: { 0.17126, 0.37524, 0.5549, 0.740604, 0.926023, 1.11109, 1.2963, 1.48148, 1.66667, 1.85185, 2.03704
, 2.22222, 2.40741, 2.59259, 2.77778, 2.96296, 3.14815, 3.33333, 3.51852, 3.7037, 3.88889, 4.07407, 4.2592
6, 4.44444, 4.62963, 4.81481, 5, 5.18519, 5.37037, 5.55556, 5.74074, 5.92593, 6.11111, 6.2963,

6.48148, 6.
66667, 6.85185, 7.03704, 7.22222, 7.40741, 7.59259, 7.77778, 7.96296, 8.14815, 8.33333, 8.51852, 8.7037, 8
.88889, 9.07407, 9.25926, 9.44444, 9.62963, 9.81481, 10, 10.1852, 10.3704, 10.5556, 10.7407, 10.9259, 11.1
111, 11.2963, 11.4815, 11.6667, 11.8519, 12.037, 12.2222, 12.4074, 12.5926, 12.7778, 12.963, 13.1481, 13.3
333, 13.5185, 13.7037, 13.8889, 14.0741, 14.2593, 14.4444, 14.6296, 14.8148, 15, 15.1852, 15.3704, 15.5556
, 15.7407, 15.9259, 16.1111, 16.2963, 16.4815, 16.6667, 16.8519, 17.0372, 17.2213, 17.4092, 17.5963, 17.73
61, 18.1074, 18.0312, 16.956, 26.4792, }

Jacobi: { 0.17126, 0.37524, 0.5549, 0.740604, 0.926023, 1.11109, 1.2963, 1.48148, 1.66667, 1.85185, 2.0370
4, 2.22222, 2.40741, 2.59259, 2.77778, 2.96296, 3.14815, 3.33333, 3.51852, 3.7037, 3.88889, 4.07407, 4.259
26, 4.44444, 4.62963, 4.81481, 5, 5.18519, 5.37037, 5.55556, 5.74074, 5.92593, 6.11111, 6.2963, 6.48148, 6
.66667, 6.85185, 7.03704, 7.22222, 7.40741, 7.59259, 7.77778, 7.96296, 8.14815, 8.33333, 8.51852, 8.7037,
8.88889, 9.07407, 9.25926, 9.44444, 9.62963, 9.81481, 10, 10.1852, 10.3704, 10.5556, 10.7407, 10.9259, 11.
1111, 11.2963, 11.4815, 11.6667, 11.8519, 12.037, 12.2222, 12.4074, 12.5926, 12.7778, 12.963, 13.1481, 13.
3333, 13.5185, 13.7037, 13.8889, 14.0741, 14.2593, 14.4444, 14.6296, 14.8148, 15, 15.1852, 15.3704, 15.555
6, 15.7407, 15.9259, 16.1111, 16.2963, 16.4815, 16.6667, 16.8519, 17.0372, 17.2213, 17.4092, 17.5963, 17.7
361, 18.1074, 18.0312, 16.956, 26.4792, }

Wyniki są jednakowe i zgadzają się z rozwiązaniem bibliotecznym.


'jacobi.txt'
'gaus.txt'
blad
iteracje
100
1
0.01
0.0001
1x10-6
1x10-8
1x10-10
1x10-12
1x10-14
1x10-16
0
20
40
60
80
100
120
140
160
180