30.11.2017r.

Dawid Ćwik
Informatyka III
grupa AP-OG

Algorytmy numeryczne
Zadanie 2
Działania na macierzach

Działania na macierzach zostały przeprowadzone przy użyciu języka Java oraz języka C++, z użyciem biblioteki Eigen. W macierzy znajdowały się losowe liczby ograniczone typem double z przedziału <0.0, 10.0>
Testy przeprowadzono dla 3 wzorow:

- $A \cdot X$
- $(A+B+C) \cdot X$
- $A \cdot (B \cdot C)$

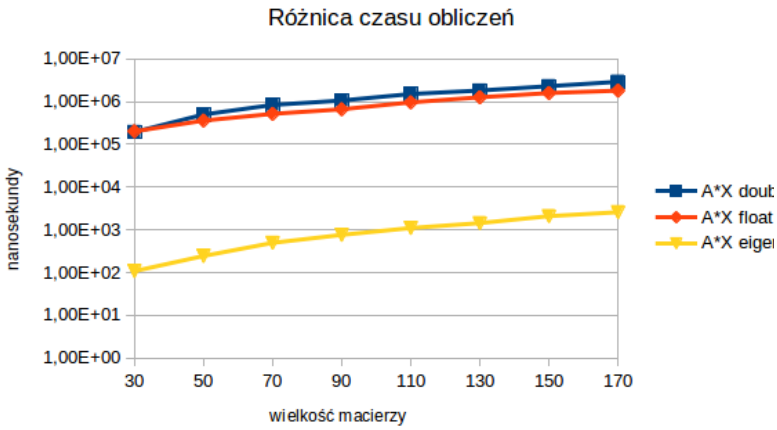
Litery A, B, C to macierze kwadratowe, a X to wektor. Użyto 3 typy zmiennych:

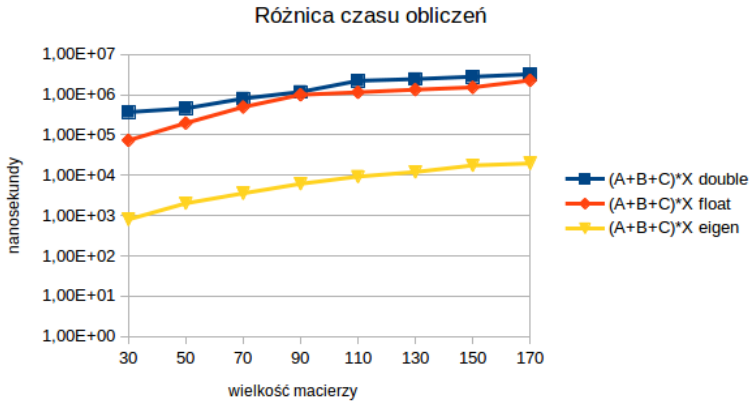
- double
- float
- typ własny (nie tracący dokładności)

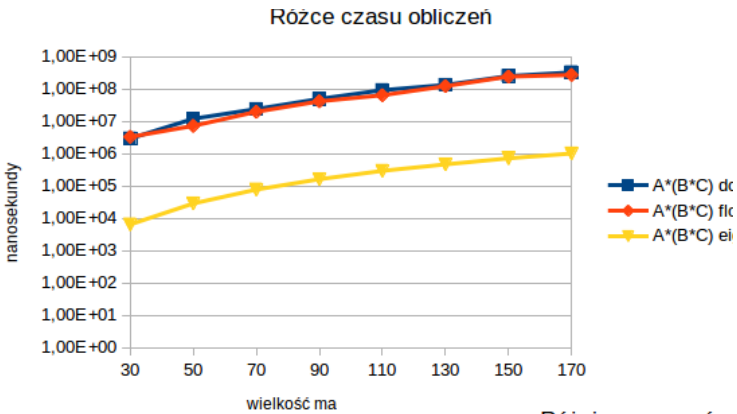
Sprawdzone został również wpływ rodzaju eliminacji Gaussa:

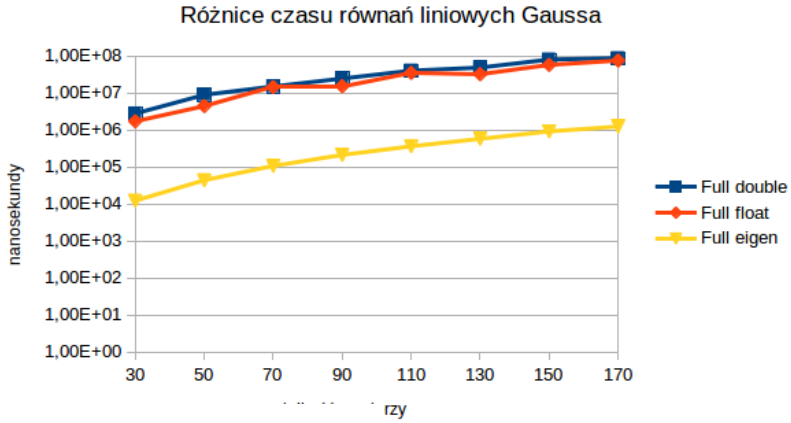
- bez wyboru elementu podstawowego,
- z częściowym wyborem elementu podstawowego,
- z peanym wyborem elementu podstawowego

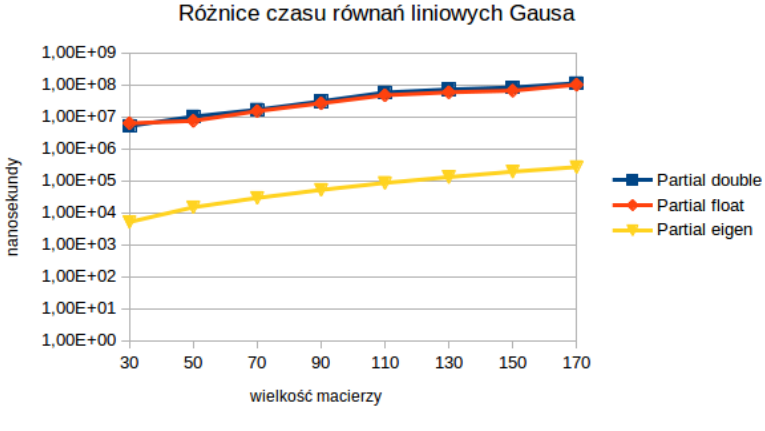
Na poniższych wykresach zaprezentowane są różnice w czasach dla 3 powyższych działań.



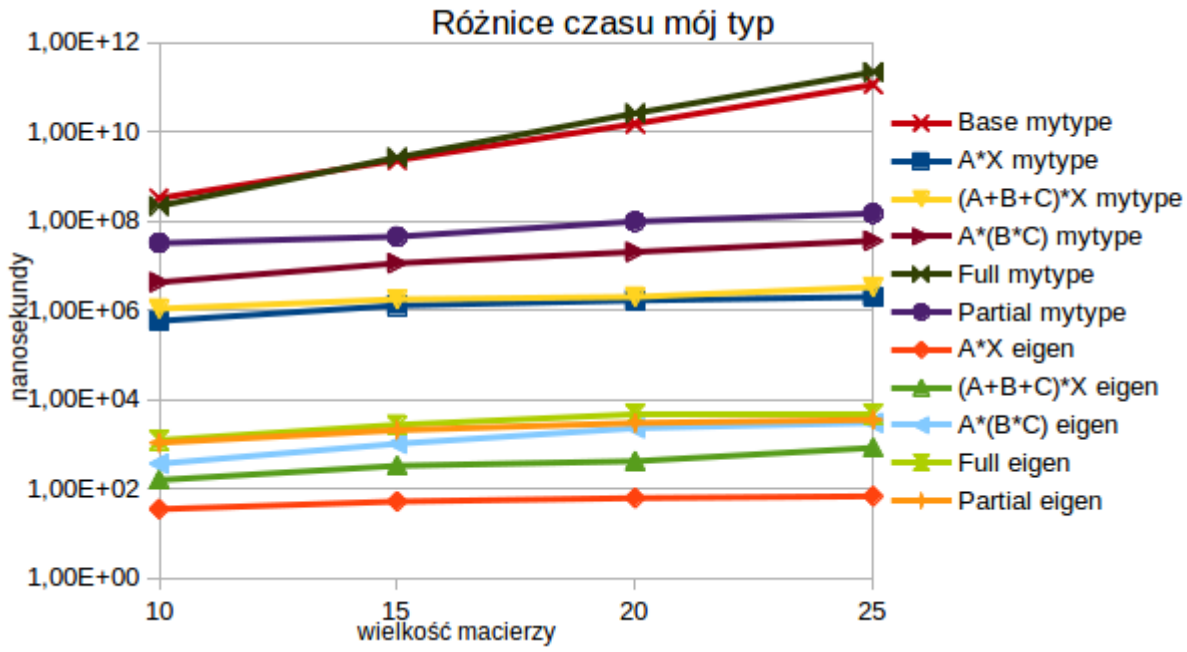








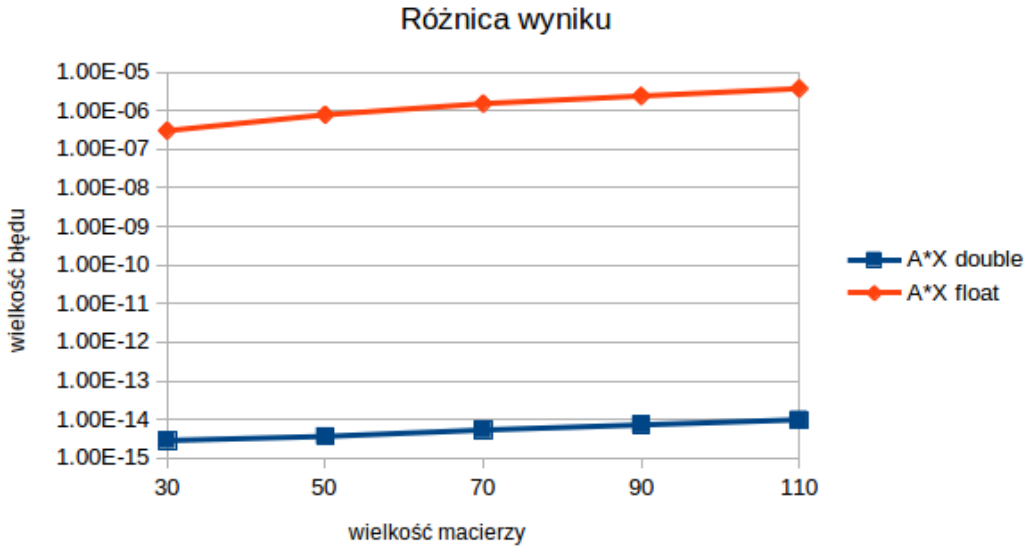
Podczas wykonywania pomiarów, zdecydownie najwolniejszym sposobem okazał się typ własny, który nie traci dokładności. Jak widzimy na wykresie poniżej, nawet dla 10 RAZY mniejszej macierzy, wyniki czasu są jeszcze większe niż dla typów double czy float. Obliczanie przy użyciu bilbioteki EIGEN pozostaje bezkonkurencyjne.



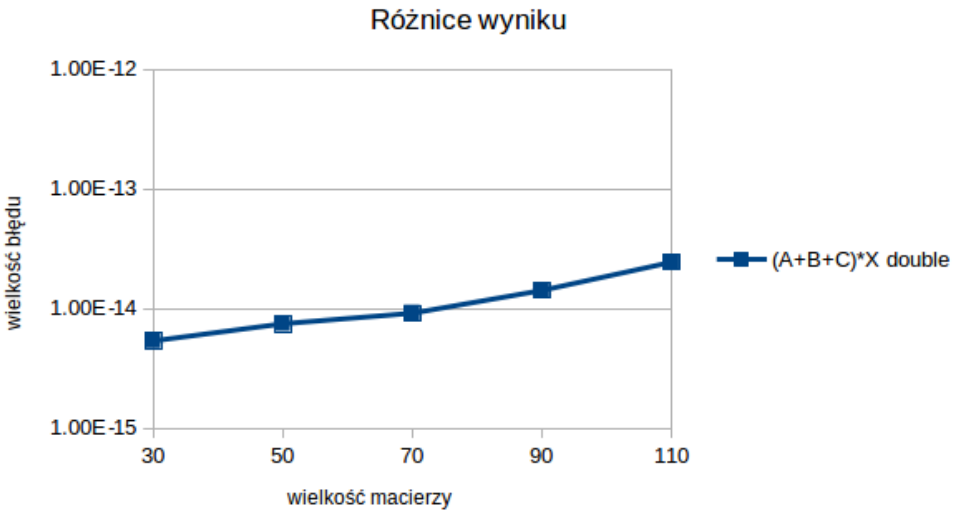
Różnice w wyliczeniach będą związane głównie z utrata precyzji przez typy, double i jeszcze mniejsze float, które po prostu „zjadaja” ostatenie cyfry co w późniejszym rozrachunku ma wpływ na wynik. Udało się temu zapobiec i uzyskać nieskończenie duże rozwinięcie ułamka, tworząc go. Poniższy screen przedstawia utworzone liczby przy wyliczaniu równania za pomocą Gausse Partial Pivoting dla macierzy 30x30.

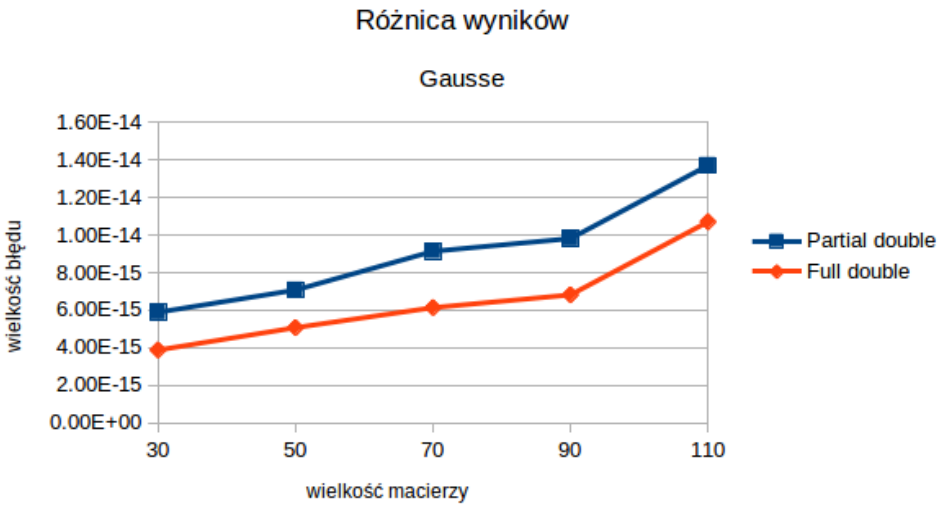
```
-3944844172138324309316707817491053824535007805540074091464897/14645586581372230053750110928502955856322099435078647041345291
3812540865542003648965833533141091186498830307660563077621/14645586581372230053750110928502955856322099435078647041345291
1102857318710519137165163752652089905498783122571442372088477/14645586581372230053750110928502955856322099435078647041345291
4103430457430595633584694111323778244685771330940481697409547/14645586581372230053750110928502955856322099435078647041345291
-1724541956925036070606147773994628273016950376565765140085757/14645586581372230053750110928502955856322099435078647041345291
1737884593124892020791023934348381276198761198898073088874417/14645586581372230053750110928502955856322099435078647041345291
5640940243855722725320442612385727290545481448930409752714817/14645586581372230053750110928502955856322099435078647041345291
-3182791500646292298174256938993805471151876302010550289654847/14645586581372230053750110928502955856322099435078647041345291
1673809108933951938177457879021386035672933388383531294817503/14645586581372230053750110928502955856322099435078647041345291
6360915430735186418962312044135139022033930518458685105227899/14645586581372230053750110928502955856322099435078647041345291
-1075089622589209461070344572105505174533066605425118014949977/14645586581372230053750110928502955856322099435078647041345291
6647467344862008441996001356832247136203403205849971683429867/14645586581372230053750110928502955856322099435078647041345291
7044773356682543599492212176414530504727839947946482096984187/14645586581372230053750110928502955856322099435078647041345291
-1692977564073025698093753675036402219974844766439556775805064/14645586581372230053750110928502955856322099435078647041345291
2125720860890830334654820894130892429132872806792249463866807/14645586581372230053750110928502955856322099435078647041345291
-6280439113263035062697200154006964018752620186017013569245177/14645586581372230053750110928502955856322099435078647041345291
-2796053907914418085973453841957477049594103464739706857802847/14645586581372230053750110928502955856322099435078647041345291
-3135488402902725615017054192333985361810115915040910076705677/14645586581372230053750110928502955856322099435078647041345291
-2591649920402012953380183879189480256590838367872988385967537/14645586581372230053750110928502955856322099435078647041345291
1134996325543922168133937144863344069194918938841722750568251/14645586581372230053750110928502955856322099435078647041345291
-1341311163057108827745728173730815488904726808881972234892977/14645586581372230053750110928502955856322099435078647041345291
-1287334505396754066853812744562918967891719223613747922077633/14645586581372230053750110928502955856322099435078647041345291
898309790552230745957104421793315510628943885678817431540077/14645586581372230053750110928502955856322099435078647041345291
3029028940488275071944364979181086214360972677110336114305397/14645586581372230053750110928502955856322099435078647041345291
4588310967117706790277216848613408708293537718683774596893827/14645586581372230053750110928502955856322099435078647041345291
7844238527221584084598549980354988427871371843792987369645187/14645586581372230053750110928502955856322099435078647041345291
7926574825023291816477792928192330185978290869380299897721277/14645586581372230053750110928502955856322099435078647041345291
1950573581178854985138106842823803634166649125522017792520527/14645586581372230053750110928502955856322099435078647041345291
-1857807819793570223363085074388013560241307042701148173635058/14645586581372230053750110928502955856322099435078647041345291
-7202482730143249585571284160315501691877184762807769206034327/14645586581372230053750110928502955856322099435078647041345291
```

Poniższe wykresy przedstawiają różnice w wynikach z wynikami uzyskanymi w bibliotece C++, EIGEN.



Jak widać na załączonym wykresie, błąd osiągano w granicach dokładności zmiennych, dla polepszenia jakości wykresów w następnych przedstawiałem wyłącznie zmienną double.





Podsumowując, biblioteka Eigen umożliwia bardzo szybkie obliczanie macierzy, przy stosunkowo bardzo dużej precyzji. Jeżeli jednak zależy nam na bardzo dużej dokładności lecz kosztem czasu, należy stworzyć własny typ ułamkowy.