Gdańsk, 08.01.2020

Algorytmy Numeryczne – Zadanie 4

Aproksymacja Profilu Wysokościowego

Kasper Cisewski 253902
Marcel Dajnowicz 253971

1. Wstęp

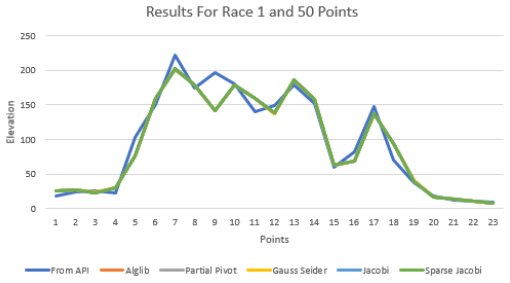
Przy pomocy 5 metod rozwiąż układ równań powstały w algorytmie CSI, które określą wysokość punktów pośrednich za pomocą aproksymacji interpolacyjnej.

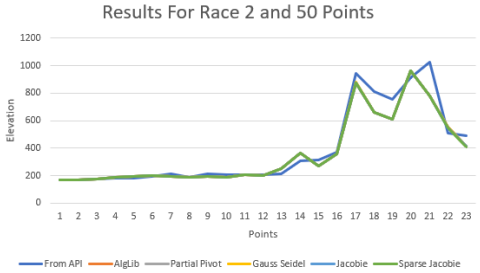
2. Dane

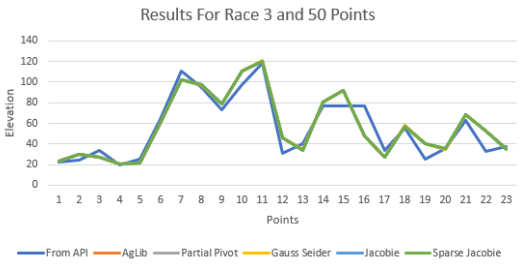
Wszystkie dane dotyczące tras zostały pobrane za pomocą Elevation API z Google Maps API. Pobraliśmy 3 etapy Tour De France, 2020 które mają najbardziej zróżnicowany profile. Race 1, która składa się z paru wzgórz, Race, 2 w którym przez połowę dominuje płaska trasa, a następnie szybkie wzniesienie i Race 3 składające się z kilku ostrych spadków i wzlotów.

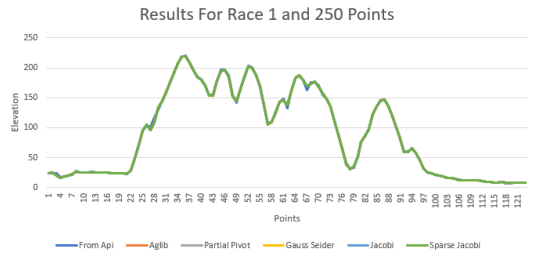
3. Testy Poprawnościowe

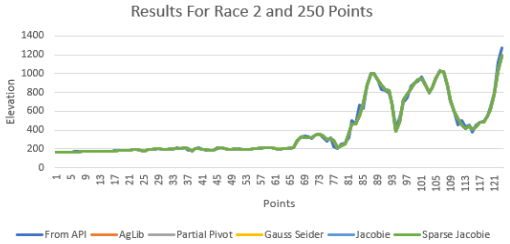
Na początku, sprawdziliśmy jak zachowują się wielomiany dla rożnej ilości węzłów, różnych tras i różnych metod. Każda z metod dawała zbliżone punkty, które uzyskaliśmy z API(Niebieska Linia), co upewniło nas w poprawności wyników. Zauważaliśmy również, że im większa liczba punktów, tym wyniki były bardziej zbliżone do tych z API. Największy błąd był zawsze, gdy wysokość gwałtowanie się zmieniała.

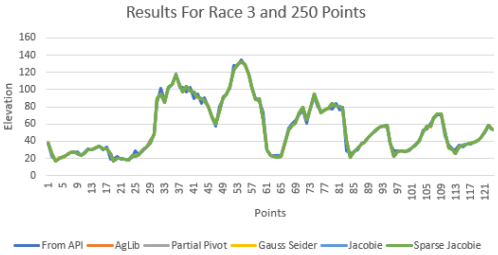


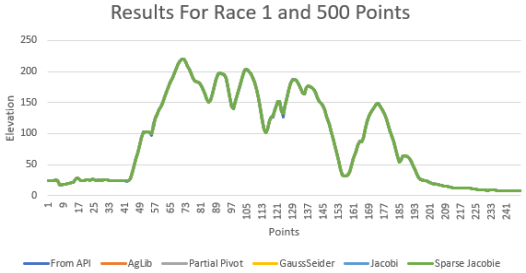


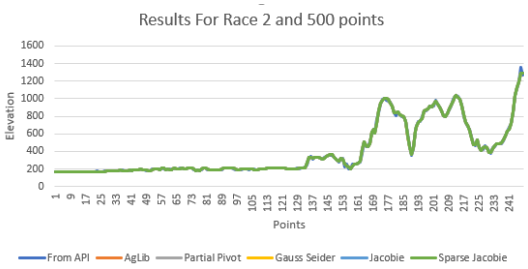


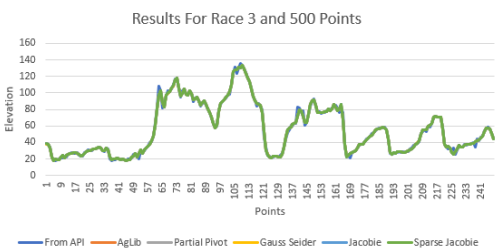




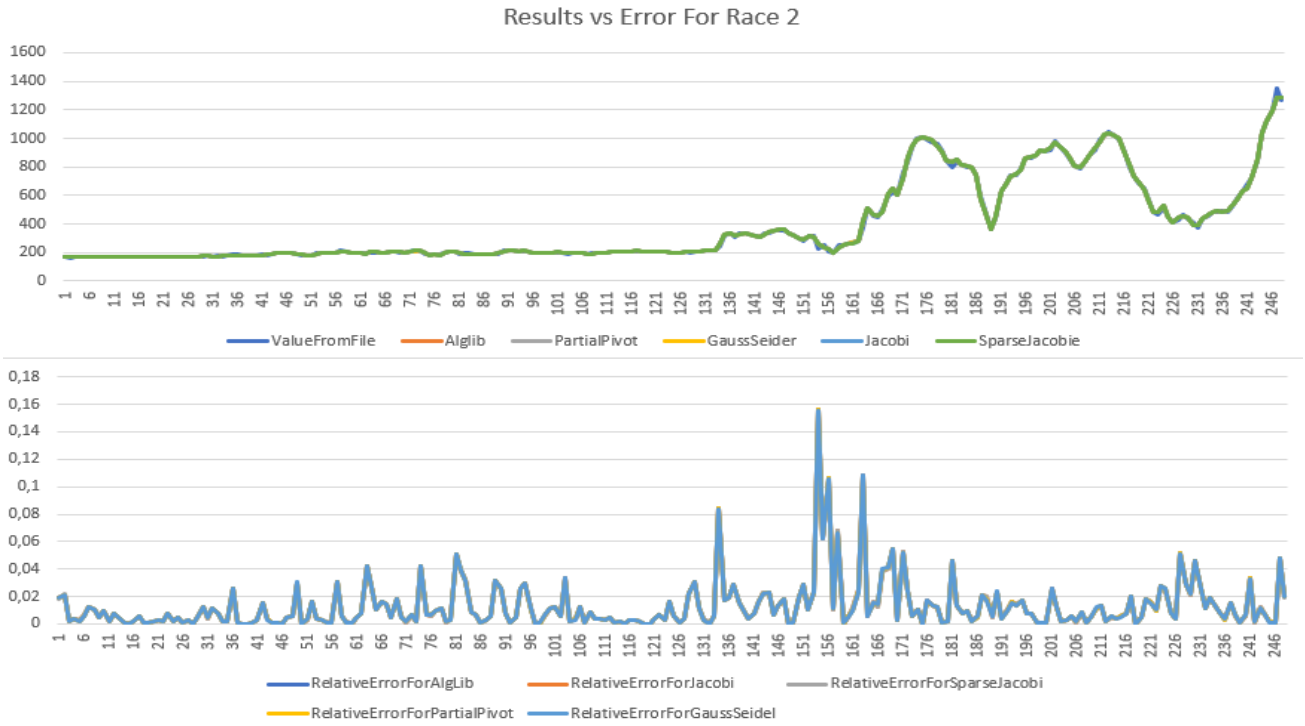








Po przeanalizowaniu błędów relatywnych ustaliliśmy, że większe błędy pojawiają się, kiedy tworzy się znacząca różnica w terenie. Żaden algorytm nie jest w stanie przewidzieć, że podczas prostej trasy nagle jest zapadnięcie się terenu. Na wykresie widać, że największy błąd uzyskaliśmy w momencie, w którym jest znaczne i nagłe obniżenie terenu.



Jak można zauważyć błędy wszystkich metod są do siebie bardzo zbliżone. Na podstawie trzech tras każdej mającej 500 punktów obliczyliśmy, jaki błąd będzie najmniejszy dla każdego punktu. AlgLib i Partial Pivot miały najwięcej razy najmniejszy błąd przez dokładne i bardziej skomplikowane obliczenia niż inne metody.

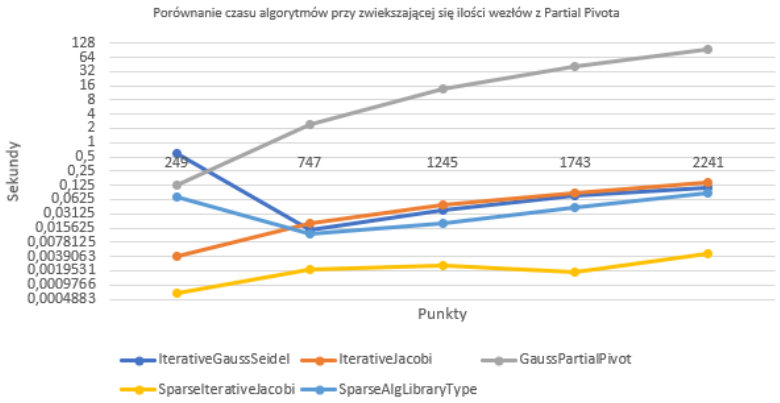
Ilość najmniejszego błędu dla każdego z punktów w trzech odcinkach dla 500 punktów

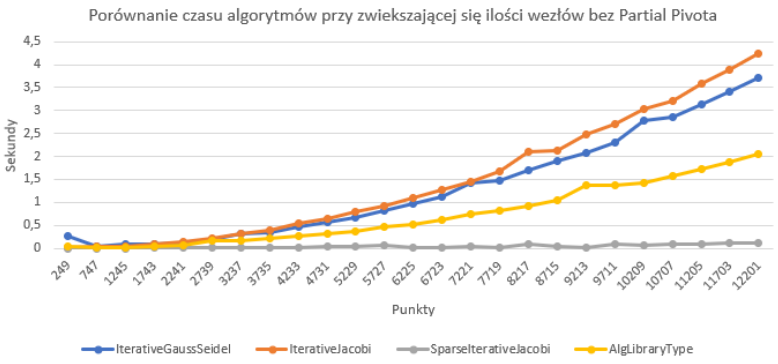
	Race 1	Race 2	Race 3
AlgLib	103	106	136
Partial Pivot	103	106	136
Gauss Seidel	80	73	65
Jacobie	64	67	45
Sparse Jacobie	64	67	45

Następnie przeszliśmy do sprawdzenia, czy obie metody iteracyjne, Gauss Seidel i Jacobie są zawsze rozbieżne. Zauważaliśmy, że metody zbiegają do siebie po około 50 iteracjach. Po zwiększaniu iteracji i korzystaniu z typu *Double*, możemy uzyskać wyniki równe metodzie PG, ale nie lepsze.

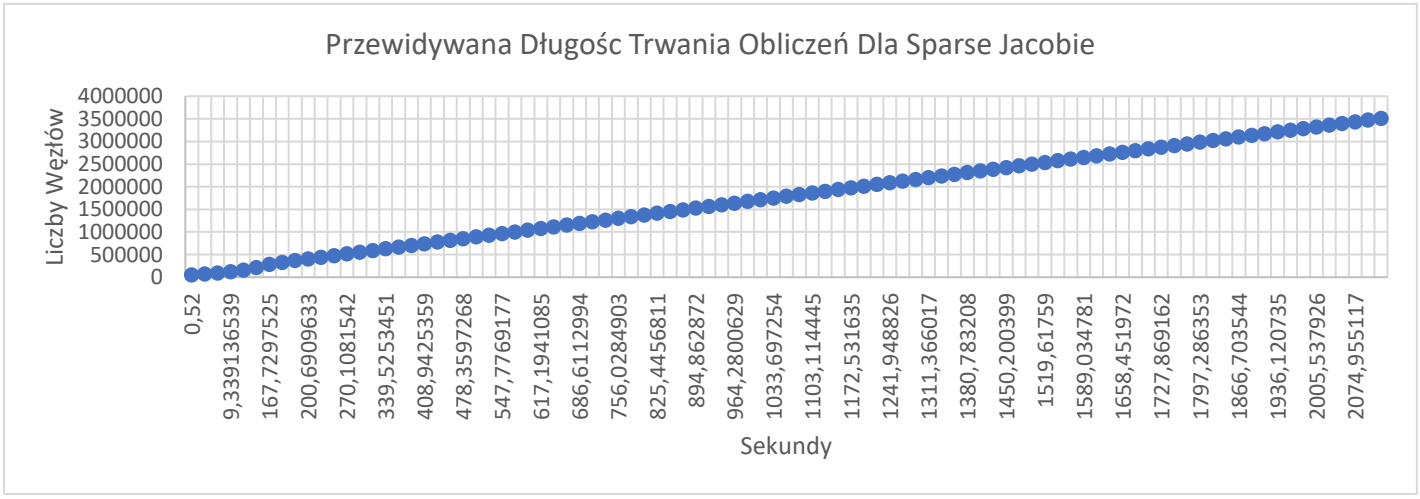
4. Testy Wydajnościowe

Na początku testów wydajnościowych, sprawdziliśmy czasy działania metod w zależności od liczby punktów. Metoda PG jest zauważalnie dłuższa przez ilość wykonywanych operacji na macierzy. Najszybszą metodą jest Sparse Jacobie przez używanie tablicy *3xP*, która trzyma cała macierz rzadką i tym samym zaoszczędza czas na kolejnych operacjach.





Następnie, zauważyliśmy, że Sparse Jacobie wykonuje operacje najszybciej. Wykorzystując nasze obliczenia oszacowaliśmy, że w ciągu 30 minut, algorytm może wykonać operacje dla 3 milinów węzłów.



Na końcu badań wydajnościowych sprawdziliśmy czas działania dla iteracyjnych algorytmów. Sparse Jacobie utrzymuje szybki czas działań, podczas gdy Gauss Seidel i Jacobie zwiększają potrzebny czas na działania wraz z ilością iteracji.