Plan pracy/zadania:

Nie musimy prosić nikogo o instalowanie źródeł/bibliotek C, pythona na serwerze MINI. Sami możemy skopiować źródła, skompilować.

1. ~~Umieścić ten dokument (plan pracy) w repozytorium do jego wglądu i edycji~~
2. Zbadać przykłady ICA jakie dane wchodzą, jakie wychodzą
3. **Wybrać odpowiednią implementację ICA w matlabie**

Wybrać implementację ICA w matlabie, dane z openfmri (czy skądkolwiek), zobaczyć wyniki, co to daje. Skonsultować z profesorem. Należy wybrać taką implementację abyśmy mieli pewność, że ona działa a wtedy dopiero możemy przechodzić do implementacji w C/C++ na CPU i CUDA.

1. **Typ danych (double czy float)** ?? Chyba raczej w matlabie jest to robione na double więc sprawdzić to jak najszybciej bo ta zmiana może podować duże komplikacjie (ilość potrzebnej pamięci). Porównać wyniki z float i double!!
2. Sprawdzić pozostałe macierze U i VT
3. Zastanowić się czy nie lepiej wywoływać funkcje z interfejsu device – wyniki z svd będziemy potrzebowali dalej w pamięci gpu (kolejne operacje na macierzach) – a tak to z interfejsem host trzeba robić kolejne transfery pamięci.

UWAGA: a może gdyby się udało zmniejszyć rozmiar m macierzy (liczbę wierszy, kosztem liczby kolumn oczywiście) to uzyskalibyśmy przyspieszenie? Wtedy jeden woksel nie mieściłby się w jednej kolumnie. Czy takie rozwiązanie jest możliwe? – zapytać się Gonzalo.

- przed tym jak spróbujesz ich zapytać, wykonaj w ogóle testy (nie zwracając większej uwagi na to czy taka operacja ma sens w świetle naszego algorytmu) czy to poprawia wydajność programu!

- zapytać ich jak im się współpracuje z lekarzami w hiszpani/ Walencji

Obliczenia mu:

size(data\_r’) = 121 x 163840

Mamy 121 wierszy i 163840 kolumn. W wyniku otrzymujemy wektor mu rozmiaru 1x163840, który zawiera średnie każdej z kolumn **data\_r’**.

Na szczęście dane trzymane są kolumnowo, więc łatwo będzie to zoptymalizować na CUDA