Anna Marciniec

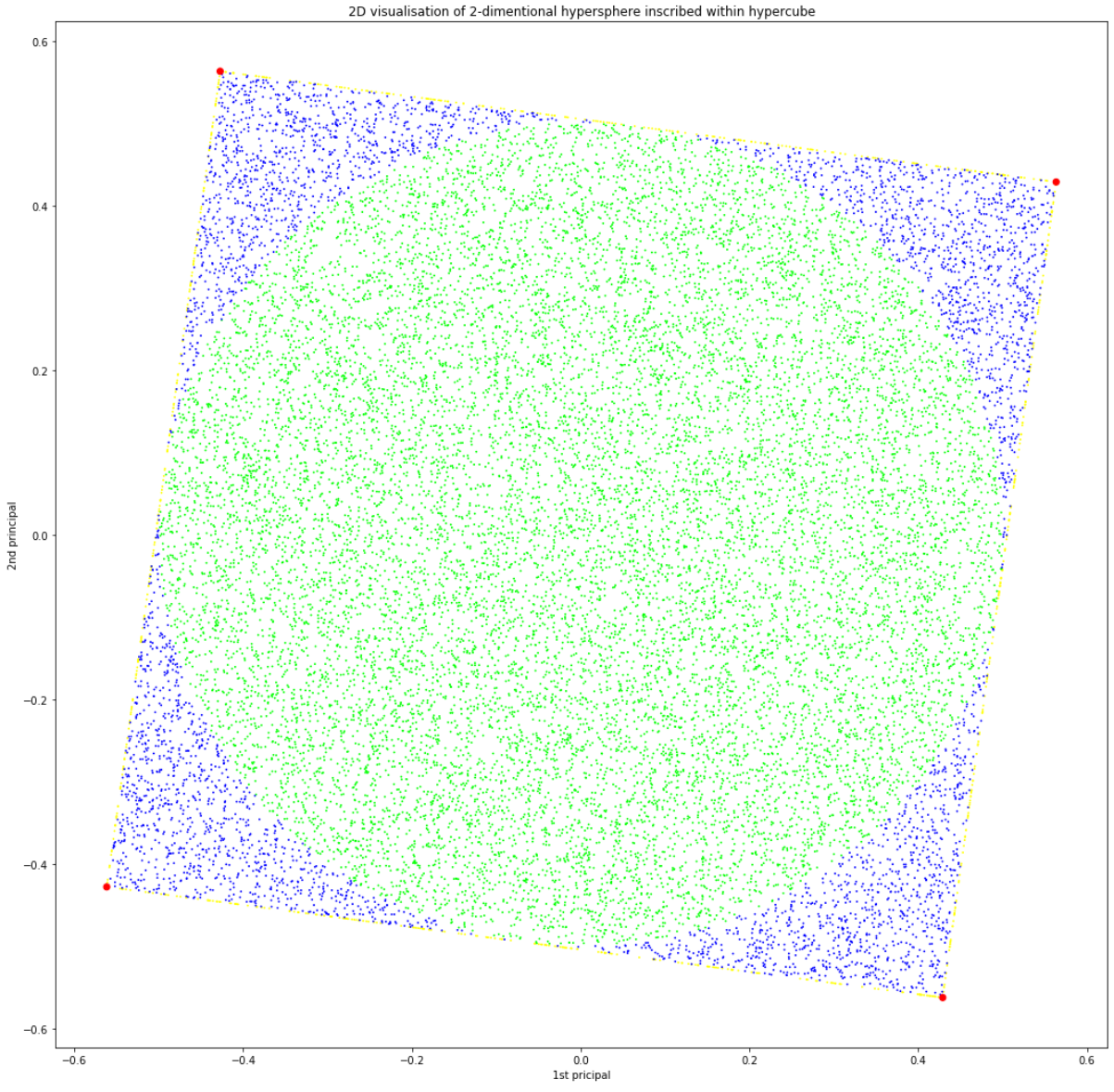
Podstawy Uczenia Maszynowego lab 3 - **PCA i kernel trick**

Raport

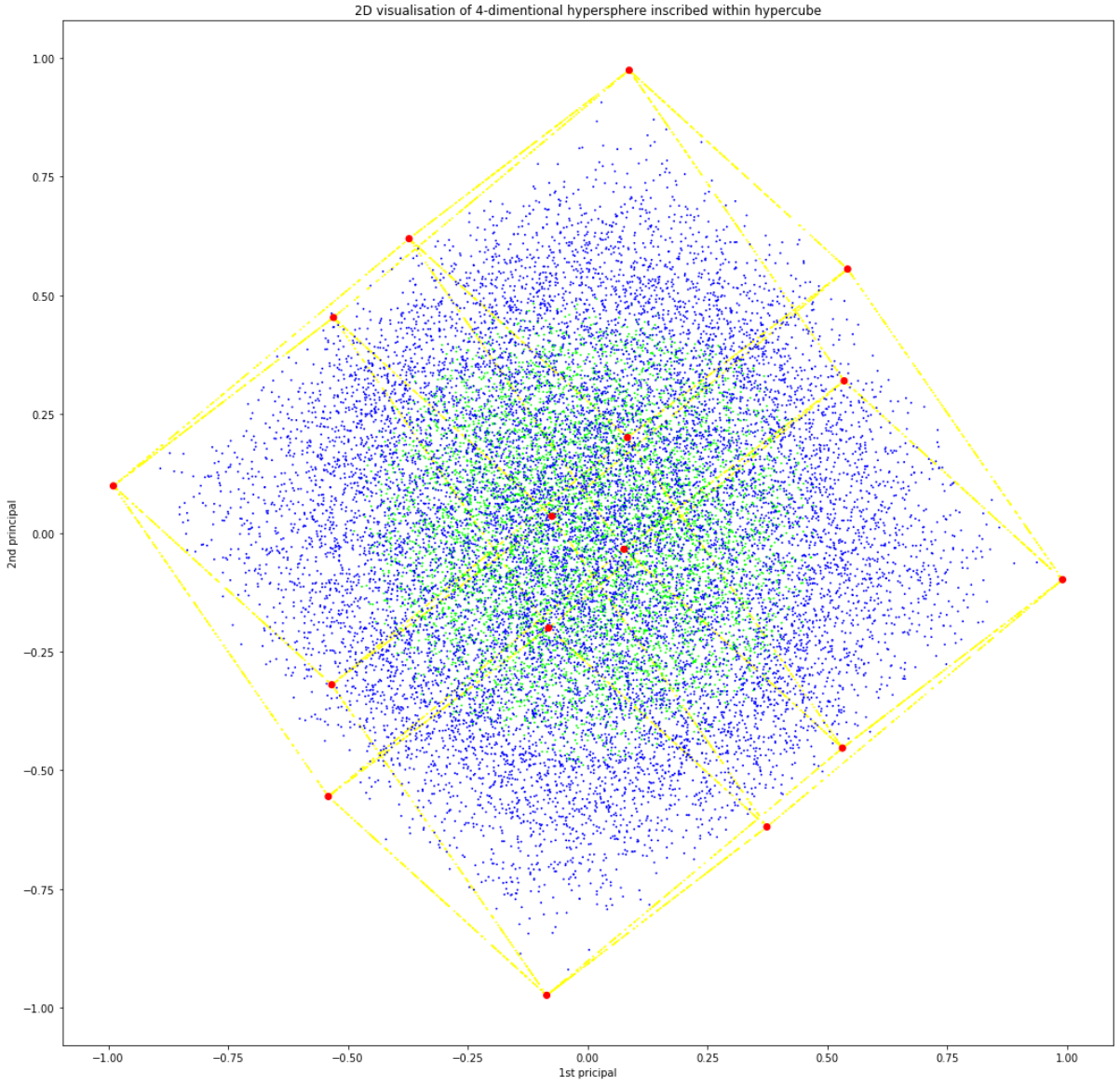
Zad A.

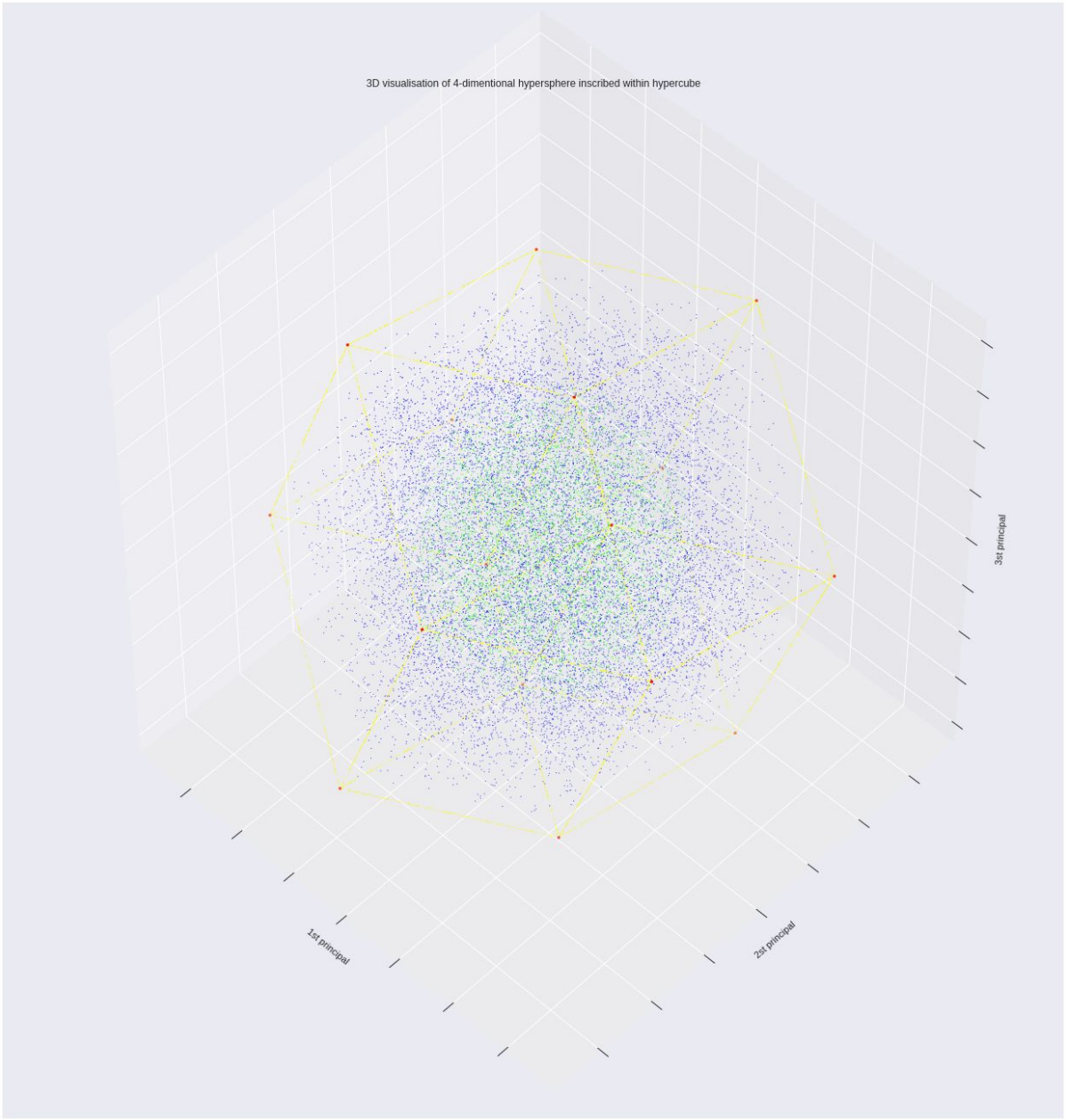
Na potrzeby tego zadania generowałam hipersześcian za pomocą 20 000 losowych punktów. Środek hipersześcianu oraz hiperkuli znajdowały się w punkcie (0,0,0,...,0), [środku układu współrzędnych]. Następnie dogenerowałam rogi hipersześcianu (w liczbie 2^n gdzie n - to wymiarowość). Dla krawędzi wygenerowałam po 100 punktów. Kolorowanie przebiegało następująco: Rogi hipersześcianu: czerwony, krawędzie: żółty, punkty w hipersześcianie, ale nie w hiperkuli: niebieski. Punkty w kuli, a nie w hipersześcianie na zielono.

Dla 2 wymiarów:

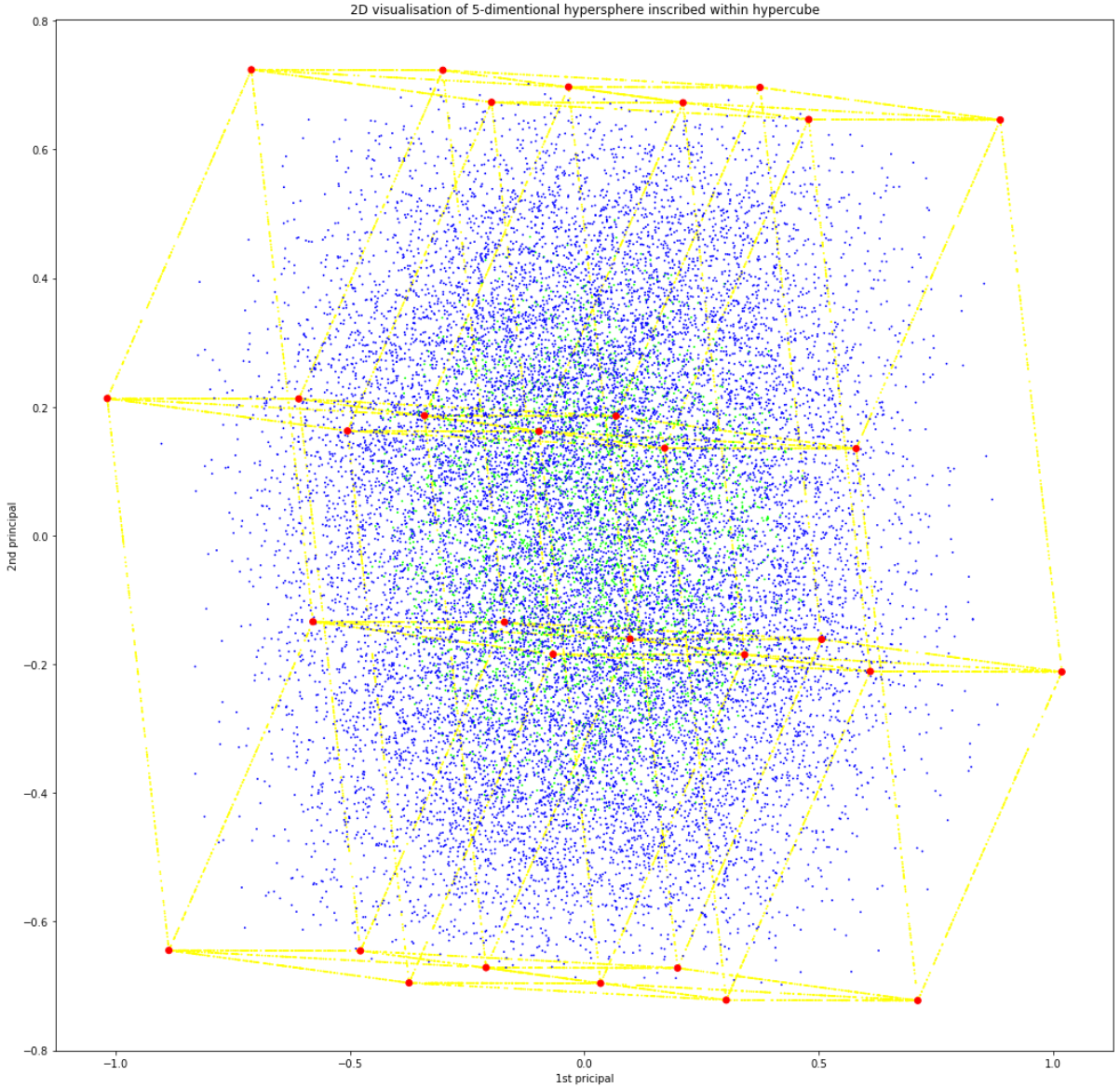


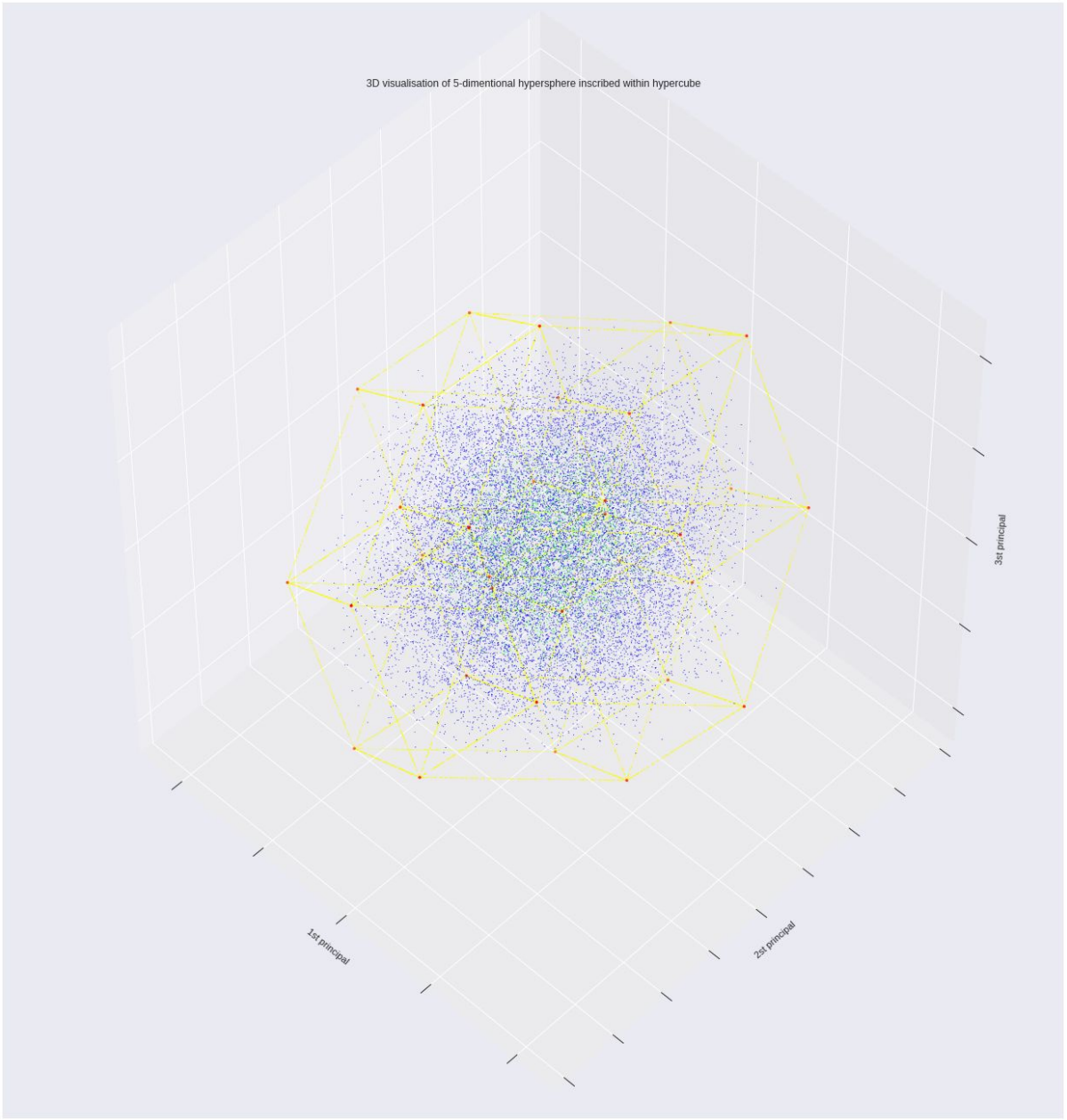
Dla 4 wymiarów:



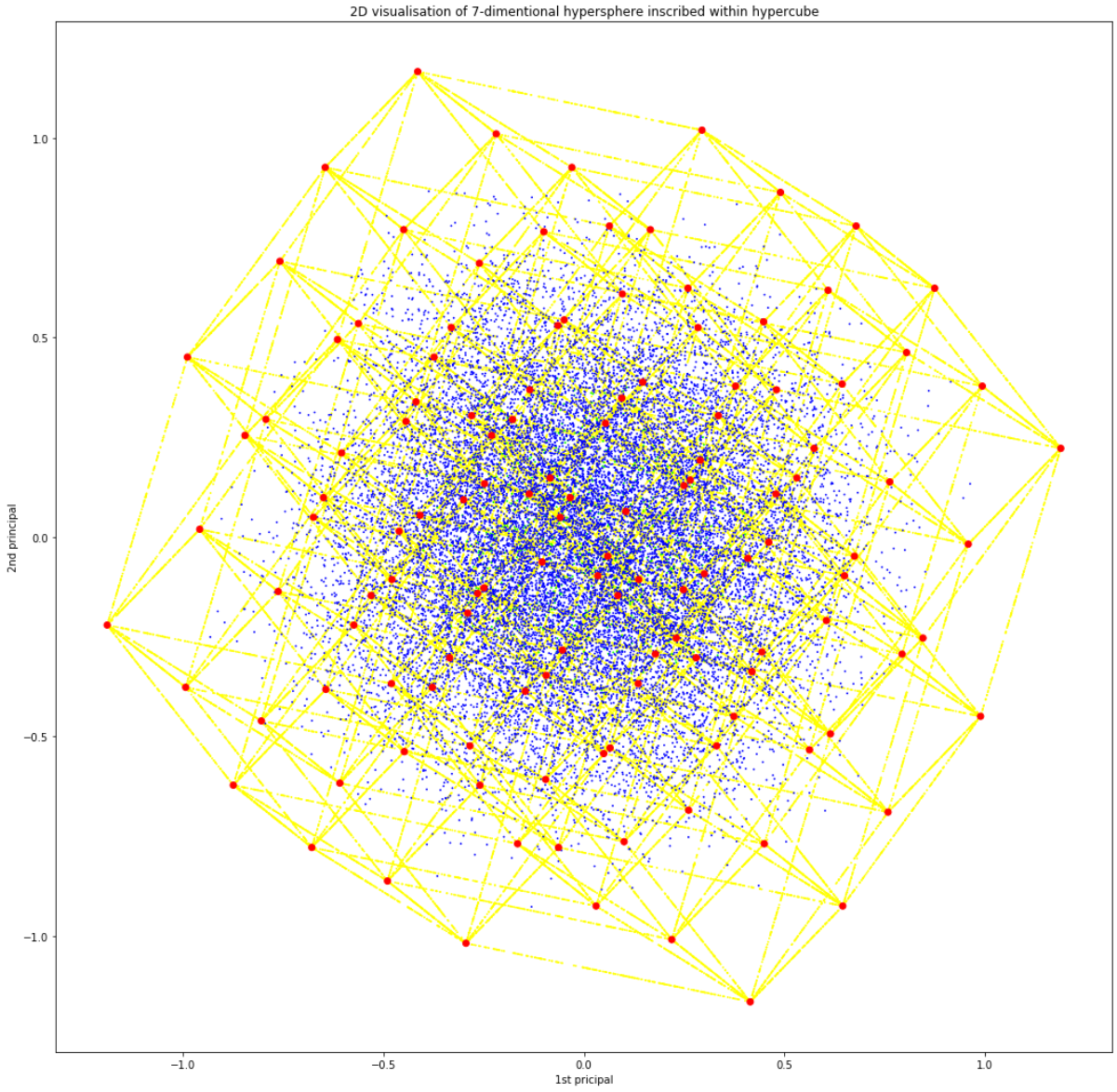

3D visualisation of 4-dimentional hypersphere inscribed within hypercube
1st principal
2st principal
3st principal

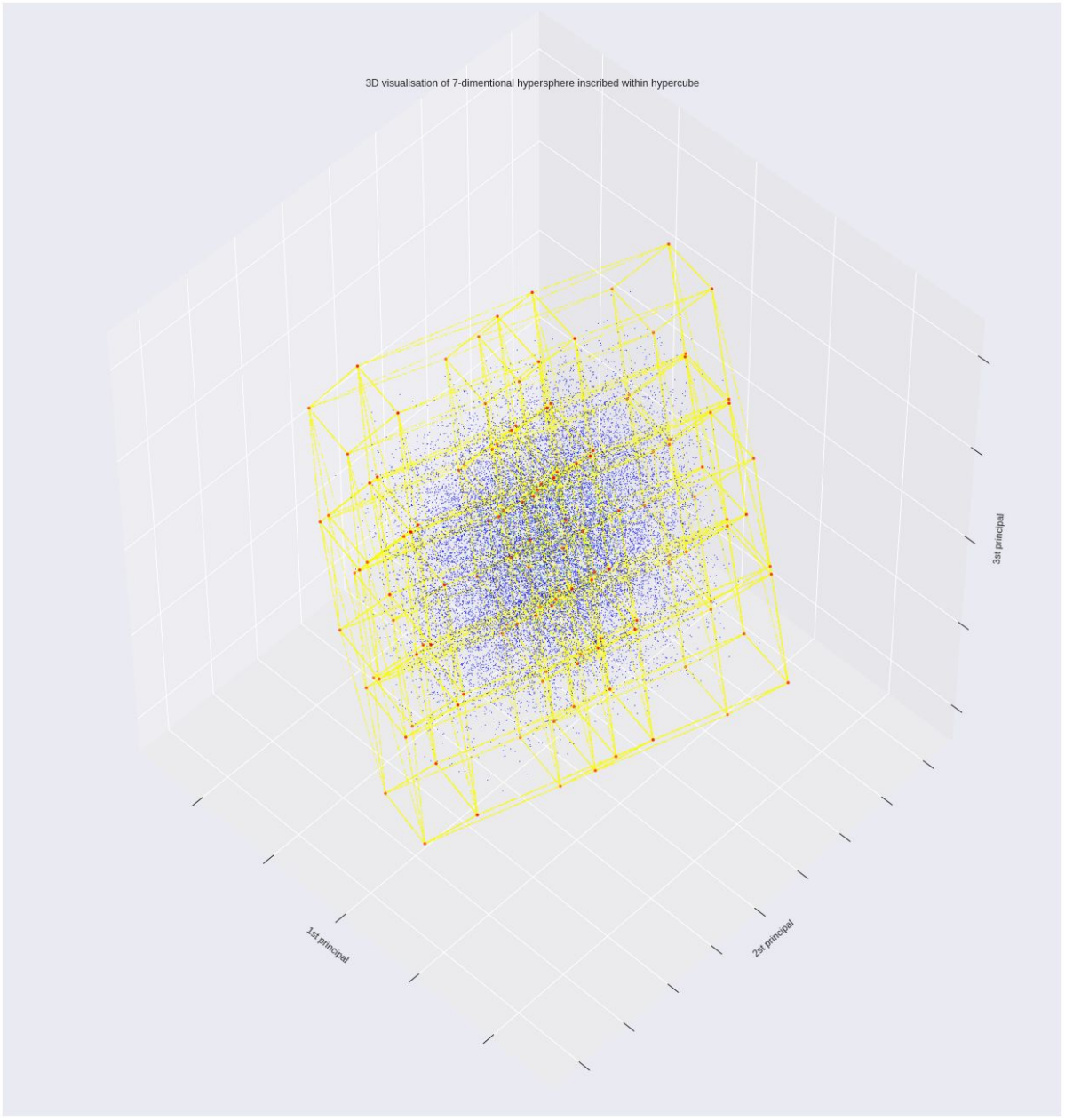
Dla 5 wymiarów:




3D visualisation of 5-dimentional hypersphere inscribed within hypercube
1st principal
2st principal
3st principal

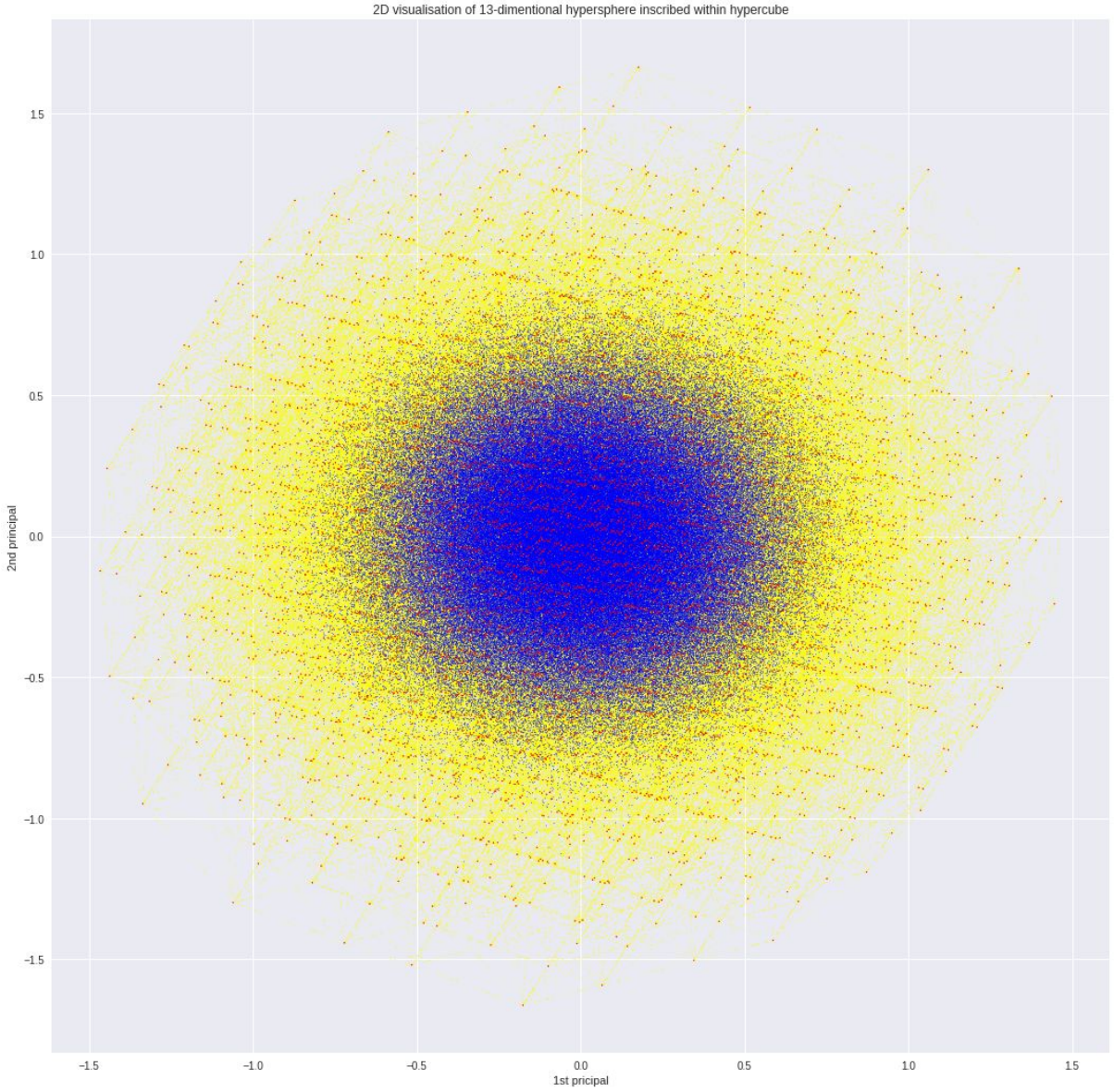
Dla 7 wymiarów:

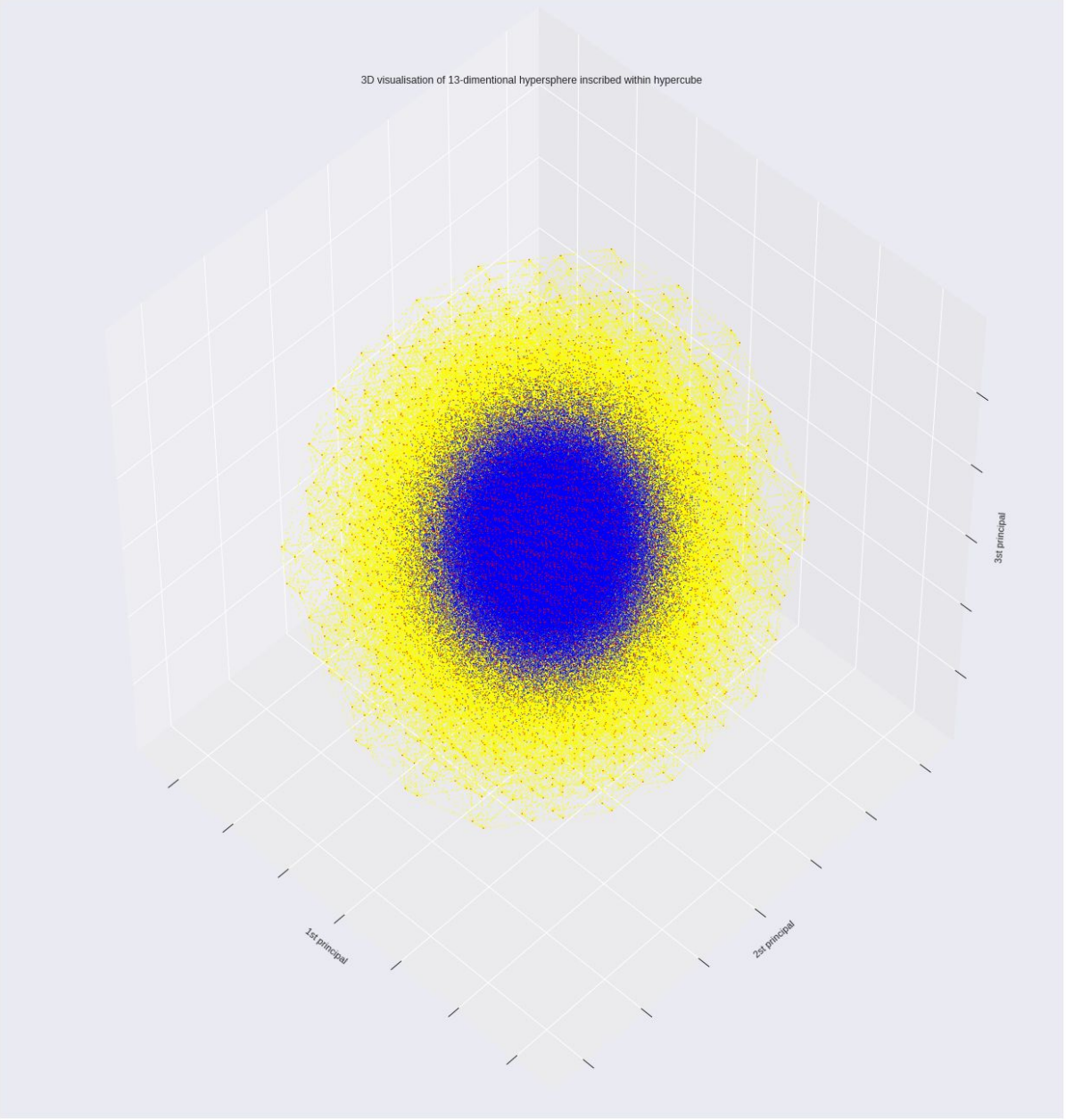




Punkty czerwone - rogi, specjalnie powiększone dla większej czytelności oglądania.

Dla 13 wymiarów zmniejszyłam liczbę generowanych punktów na krawędziach do 10 i liczbę punktów wewnątrz hipersześcianu zwiększyć do 300 000.





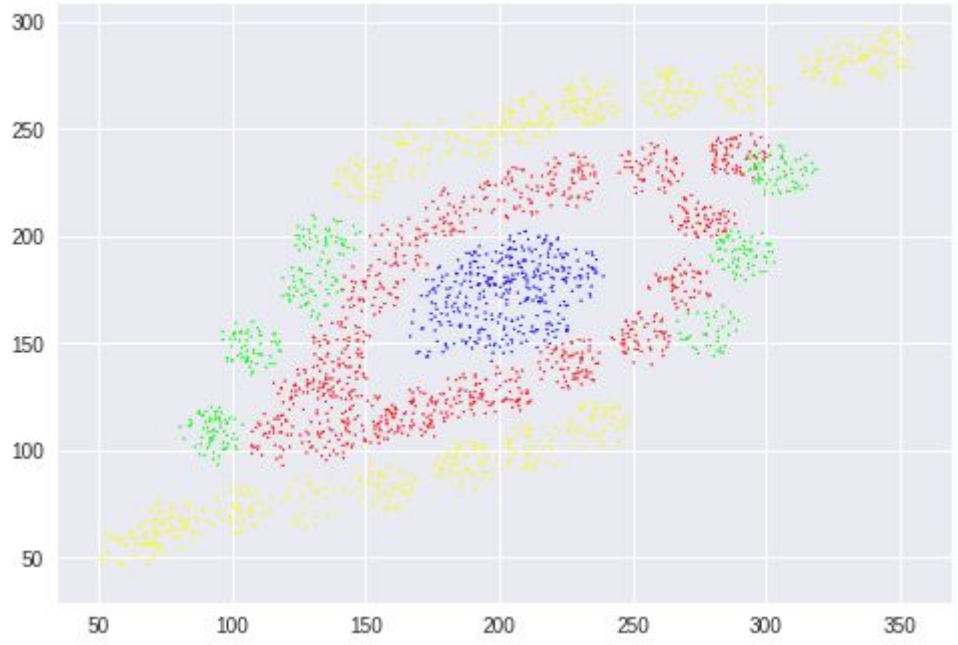
Wnioski:

- Można zauważyć, że dla 2 wymiarów, hipersześcian został przekręcony. Nowy wektor bazowy przekręca i rozciąga.
- Im więcej wymiarów tym gorzej wyglądają wykresy po transformacji. Dla 13 wymiarów wykresy są bardzo nieczytelne, należało zmniejszyć liczbę generowanych punktów na krawędziach i zwiększyć punkty w hipersześcianie.
- Zbiór danych został zmniejszony, mamy 2 i 3 wymiarowe wektory zamiast wielowymiarowych, więc jakbyśmy je chcieli przetwarzać później to jest szybciej .

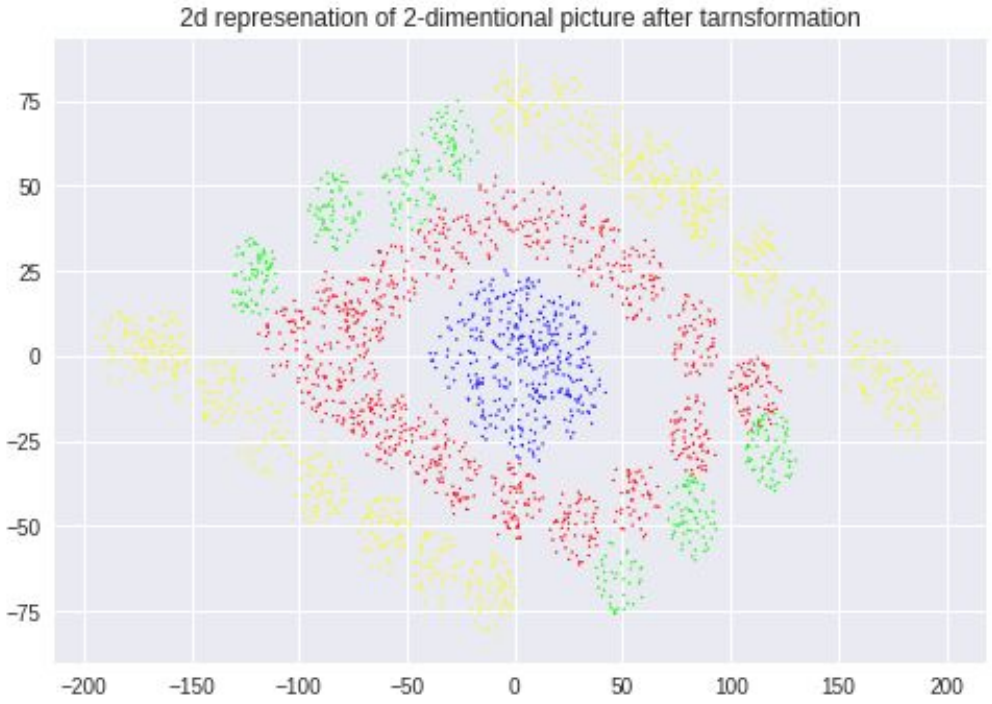
Zad B.

Do tego zadania skorzystałam z załączonego obrazka.

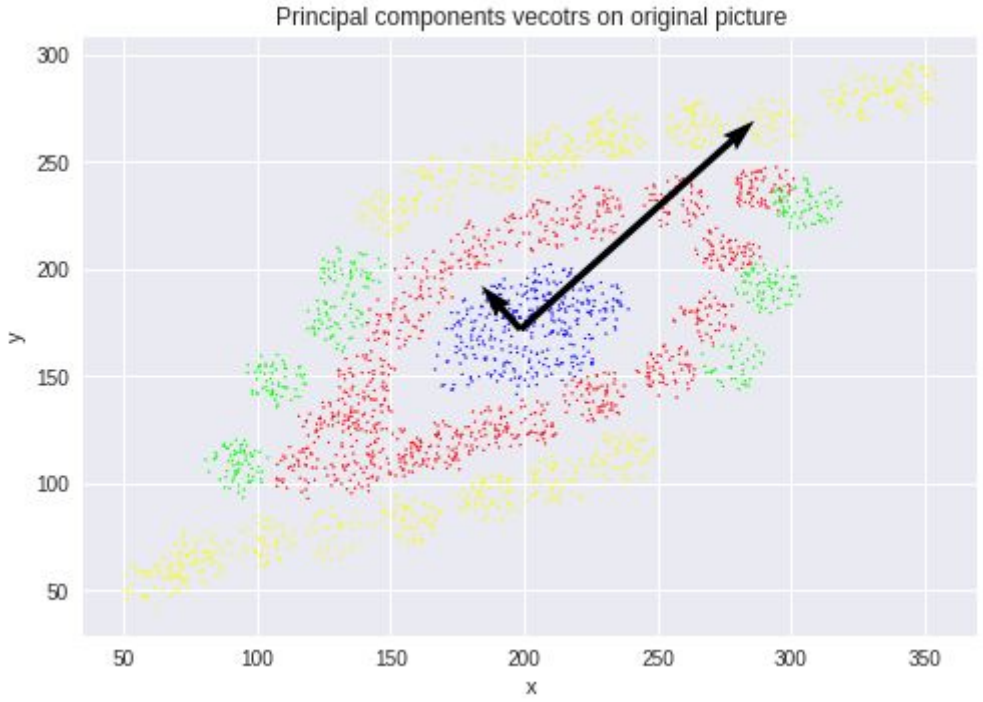
Oryginalny zbiór danych:



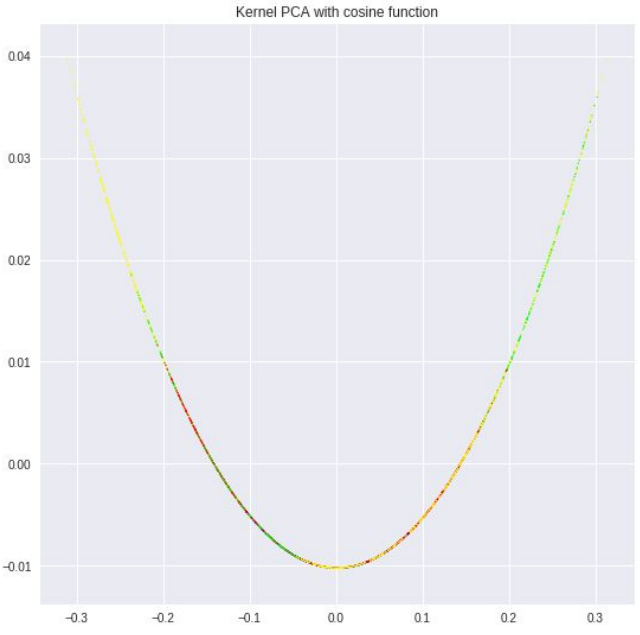
Po przekształceniu:



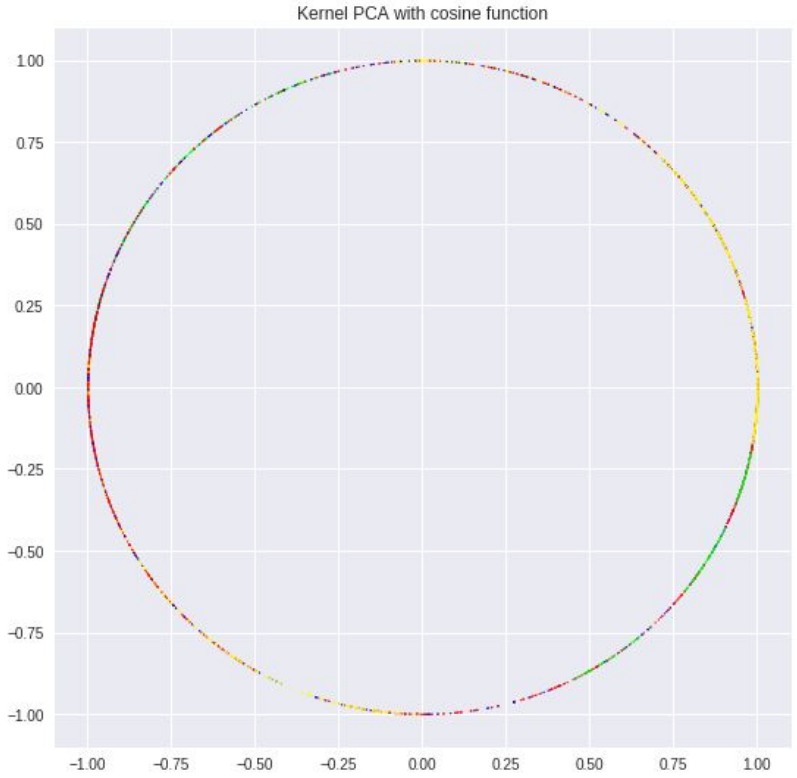
Obrazek z nałożonymi wektorami “principal components”. Długość wektora proporcjonalna do wariancji przez niego wyjaśnionej. Wektory przeskalowane, aby było przejrzyście.



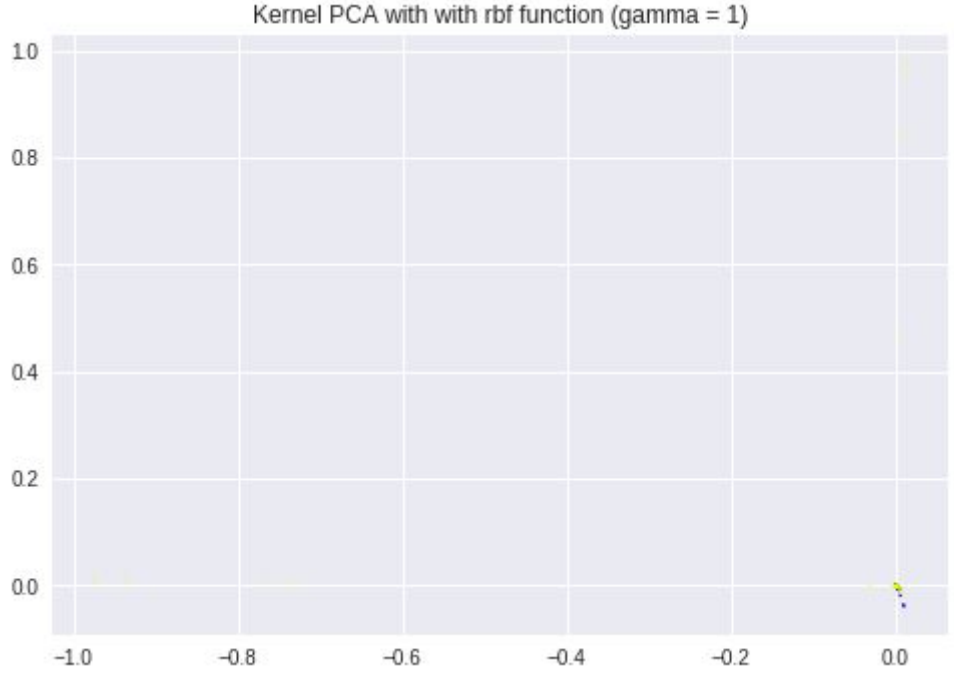
Kernel PCA:
Cosine kernel przed wyśrodkowaniem

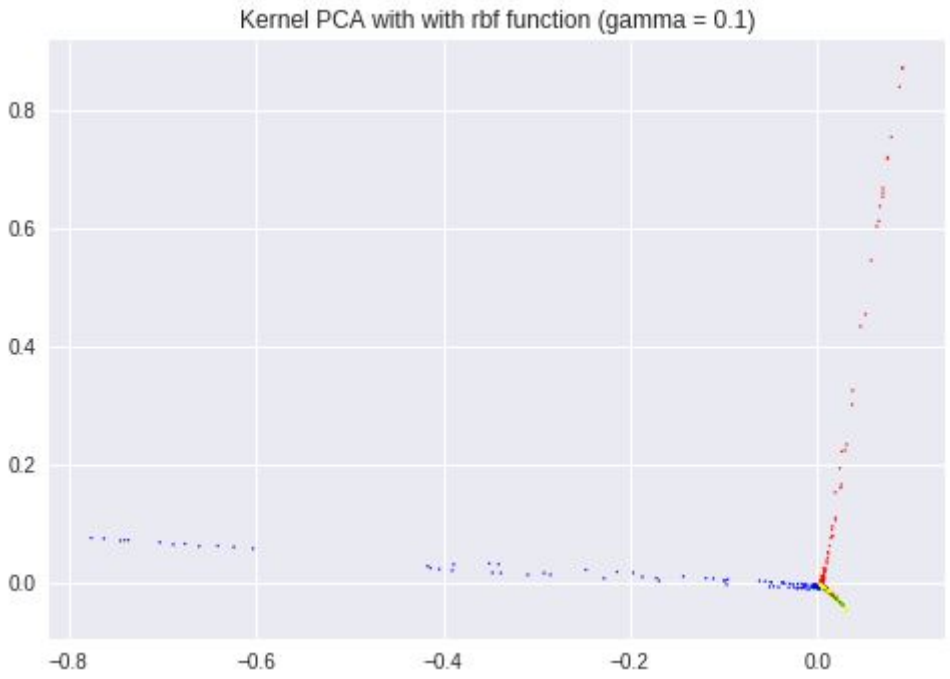


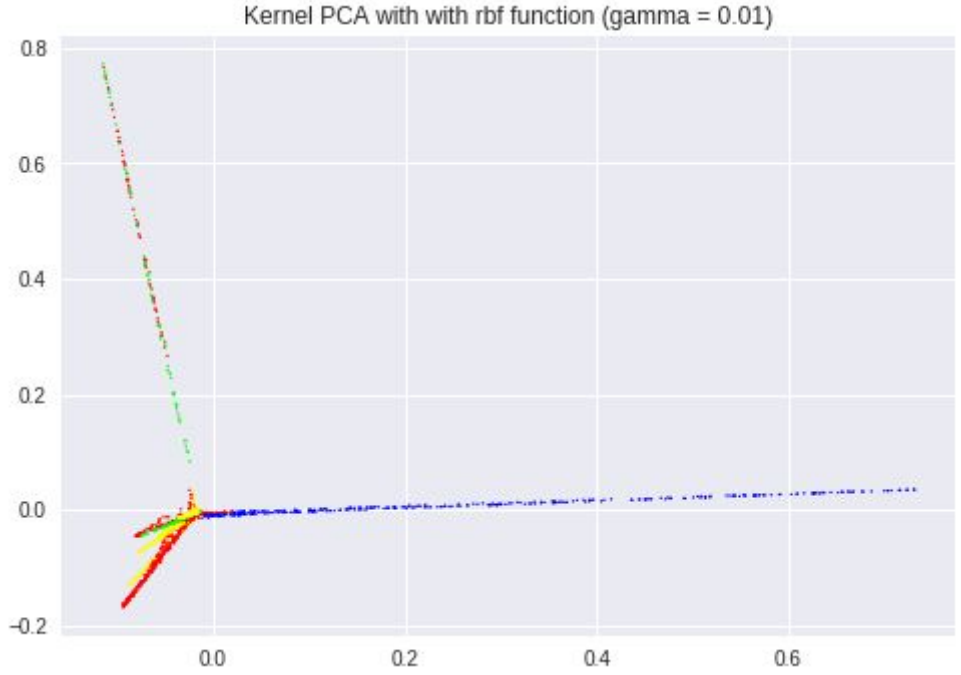
Cosine kernel po wyśrodkowaniu.

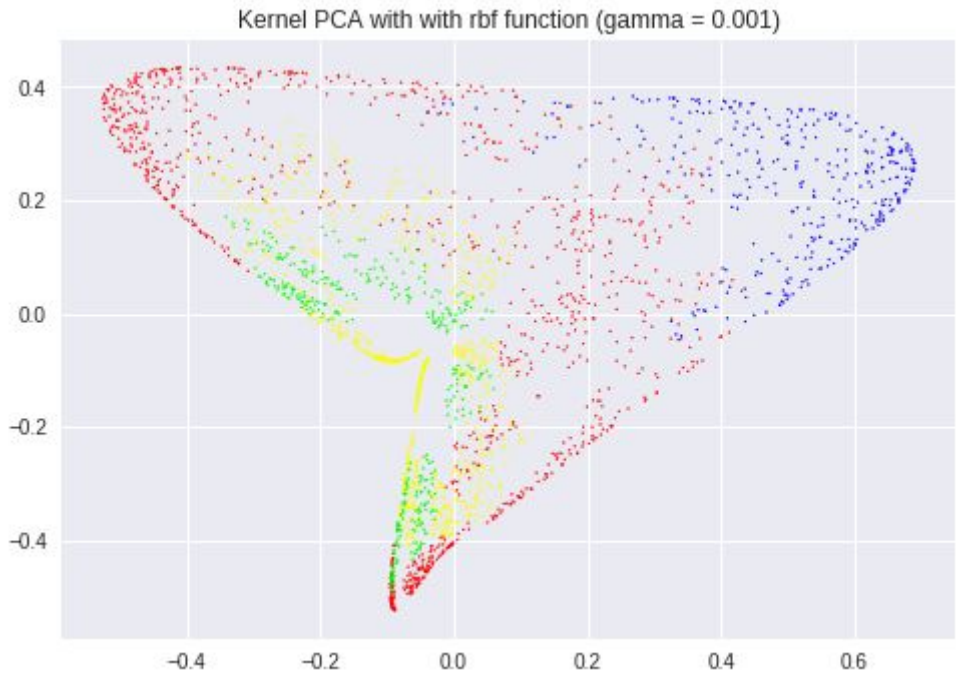


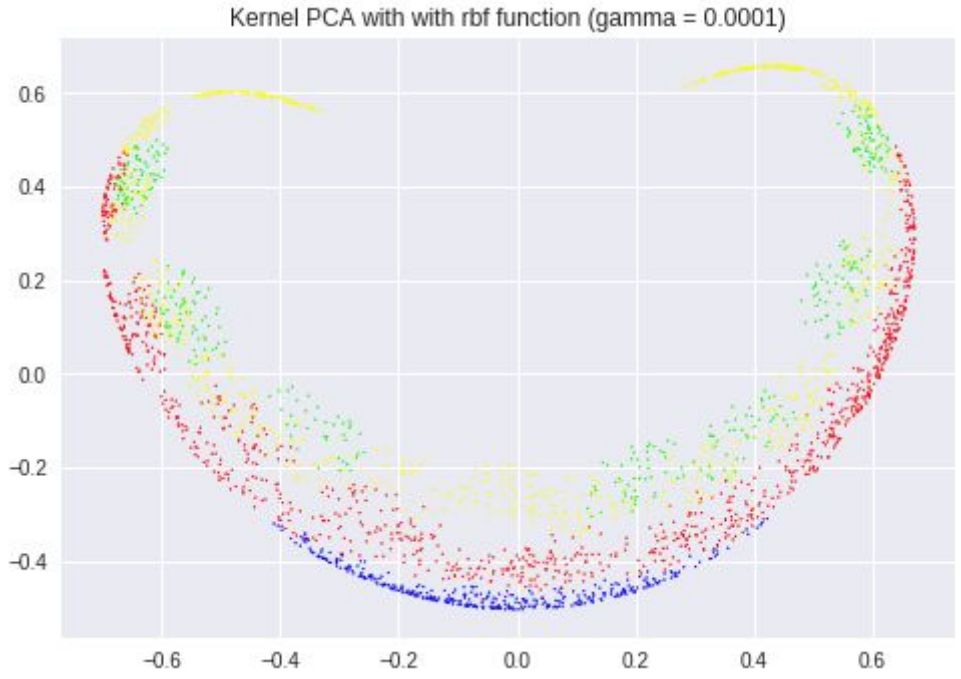
radial basis function kernel w zależności od wartości od parametru gamma:

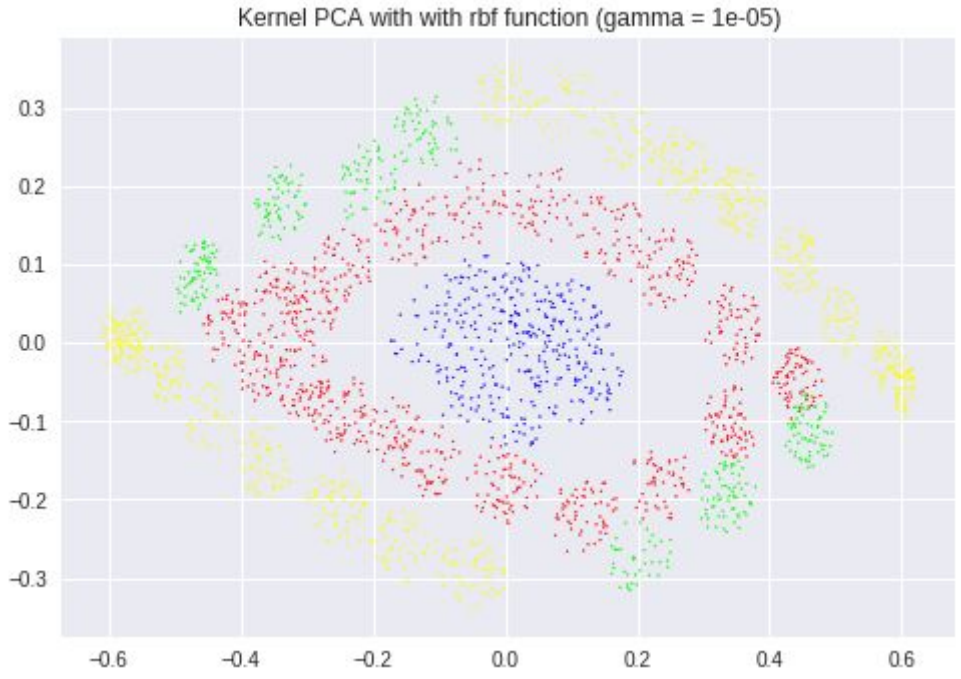


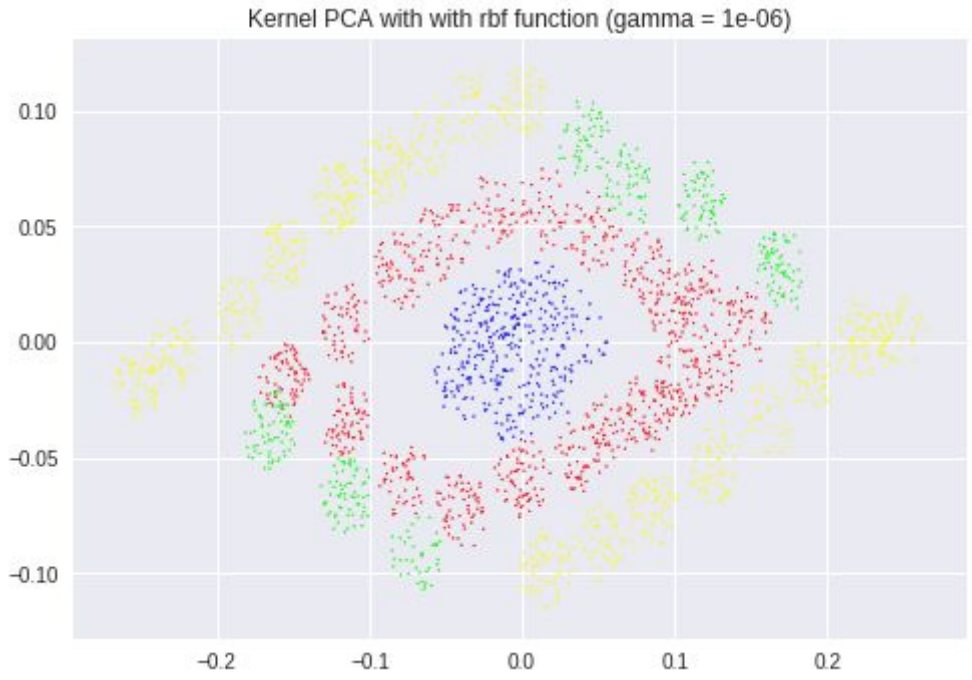

Kernel PCA with with rbf function (gamma = 0.1)
0.8
0.6
0.4
0.2
0.0
−0.8
−0.6
−0.4
−0.2
0.0

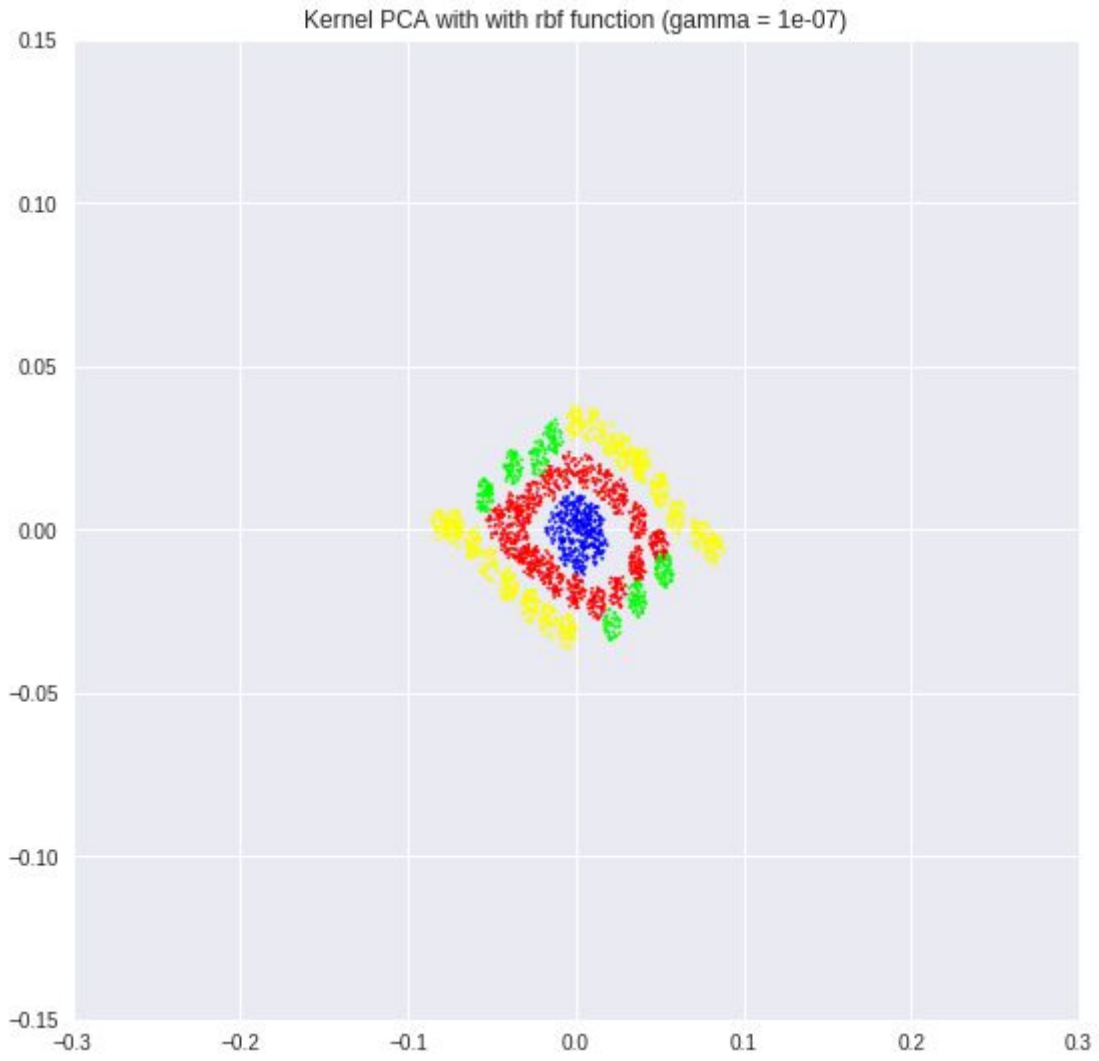

Kernel PCA with with rbf function (gamma = 0.01)
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6


Kernel PCA with with rbf function (gamma = 0.001)
0.4
0.2
0.0
−0.2
−0.4
−0.4
−0.2
0.0
0.2
0.4
0.6


Kernel PCA with with rbf function (gamma = 0.0001)
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6


Kernel PCA with with rbf function (gamma = 1e-05)
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6


Kernel PCA with with rbf function (gamma = 1e-06)
0.10
0.05
0.00
−0.05
−0.10
−0.2
−0.1
0.0
0.1
0.2



Wnioski:

- po zastosowaniu zwykłego PCA, z równoległoboku powstał rąb. Baza została zmieniona tak, że przewróciła obraz oraz go rozciągnęła.
- przy użyciu funkcji cosine przed wyśrodkowaniem, otrzymujemy wykres(hiperbole) w którym na górnych końcach widać kolor, które są daleko od środka obrazka, zielony i żółty, ale im bliżej ekstremum kolory się mieszają, natomiast po wyśrodkowaniu, otrzymujemy okrąg w którym kolory się przemieszały
- Funkcja RBF dla bardzo małych wartości gamma działa podobnie do PCA(zmienia wielkość i obraca) przy współczynniku 1e-7 mamy zmniejszony obraz w porównaniu ze współczynnikiem 1e-6.
- Dla współczynnika gamma = 1 dla tych danych na wykresie widać bardzo niewiele, wraz z maleniem współczynnika widać separację kolorów, współczynniki 0,1 i 0,01 bardzo dobrze obrazują separacje, tworząc “linie”, dla współczynnika 0,01 mamy “warstwy” ale widać że czerwony odseparowuje niebieski od zielonego i żółtego.