Praca Domowa 2 Testy

Jan Borowski

Wstęp

W poniższym raporci przedstawię wyniki testów mojej implementaci algorytmu spectralnego na stworzonych do tego celu 3 zbiorach testowych. Poniżej kod implementacij :

```
Mnn <- function(X,M){
  #'Przymuje macierz lub data.frame
  #'Funkcja wyznacza M najblirzszych sąsiadów punktów podanych w
  #'formacie macierzy gdzie mjesce i,j to j-ta współrzędna i-tego punktu.
  #'Zwraca macierz gdzie miesce i,j to j-ty indeks sonsiad i-tego punktu.
  n <- length(X[,1])
  X <- as.matrix(dist(X,method = "euclidean",upper = TRUE))
  X <- apply(X,1,order)
  X<- t(X[2:(M+1),])
  X
}
Mnn_graph <- function(S){
  #' Funkcja zwraca graf sąsiedstwa dla wygenrowanej wczesniej macierzy  sąsiedstwa
  # Używam pakietu igraph
  n <- nrow(S)
  m <- ncol(S)
  # Tworzę dwie ramki z jedna kolumną zawierajacą numery wierzchołków
  # a drugą numery jednego z ich sąsiadów (wierzchołki powarzają się tyle
  # razy ile najbliższych sąsiadów wyznaczono)
  vartex1 <- rep(1:nrow(S), each = m)
  nigerbous1 <- unlist(as.data.frame(t(S)))
  edges <- cbind(vartex1, nigerbous1)
  edges_symetrical <- cbind(vartex1, nigerbous1)
  # Używam powstałych ramek do stworzenia 0,1 macierzy sąsiedstwa
  G <- matrix(0, nrow = n, ncol = n)
  G[edges] <- 1
  G[edges_symetrical] <- 1
  # korzystam z pakietu igraph aby z powstałej macierzy stworzyć graf
  G<- graph_from_adjacency_matrix(G,mode = "undirected")
  # dodaję krawędzie do grafu aby był on spójny
  if (!count_components(G)==1){
  g <- !duplicated(clusters(G)$membership)
  c <- (1:n)[g]
  if (length(c)>2){
    d <- c[2:(length(c)-1)]
    d <- rep(d,each=2)
    c <- c(c[1],d,c[length(c)])
  }
  G<- add_edges(G,c)
  return(G)}
  return(G)
}
Laplacian_eigen <- function(G,k){
```

```
  #'Funkcja przyjmuję graf sąsiedstwa zwraca k wektorów odopwiadajacyh najmniejszym wartościom
  #'własnym laplasjanu tego grafu.
  # Korzystam z wbudowanej funkcij pakietu igraph
  # do wyznaczenia laplasianu
  # warto zauważyć ,że funkcij zwraca macierz rzadką
  L <- laplacian_matrix(G)
  L <- eigen(L)
  n <- length(L$value)
  g <- L$vectors[,(n):(n-k+1)]
  # Zdecydowałem się urzyć k najmnijszych
  # ponieważ z testów wynika lepsze działanie w takim wypadku
  g
}
Spectral_algoritm_cluster <- function(X,M,k){
  #'Funkcja wyznacza podział zbioru na k grup
  #'używjac algorytmu spektralnego
  #' M = liczba najbliższych sąsiadów
  W <-  Mnn(X,M)
  D <- Mnn_graph(W)
  Z <- Laplacian_eigen(D,k)
  wynik <- kmeans(Z,k)
  wynik$cluster

}
```

Testy wykonywanę są dla paremetru **k** ustalonego przy tworzeniu zbioru oraz dla parametru **M** z przdziału od **2** do **k** +10 .

Wynik algorytmu bedzie porównywany z domyślnym podziałem za pomoacą skorygowanego indeksu Randa,indeksu Fowlkesa–Mallowsa oraz wygodnej graficznej reprezentacij. Zarówno współczynniki jakościowe jak i podział bedą prezentowane dla najlepszego **M**.

Zbiór 1

Pierwszy zbiór to dwie figruy 2D zaburzone o szum losowy. Ma on na celu pokazać skuteczność mojej implementaci algorytmu spektralnego na zbiorze z którym nie poradził by sobie algorytm k-średnich. Przedstawiony na wykresie wygląda następująco:

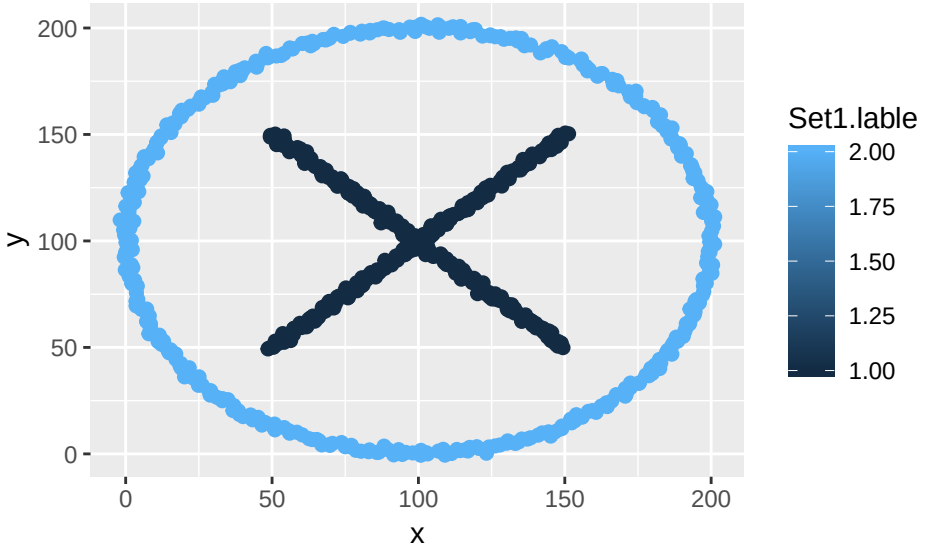

Figure 1: Zbiór 1 podział domyślny

Podział pozostaje dobry nawet przy zmiennym **M**:

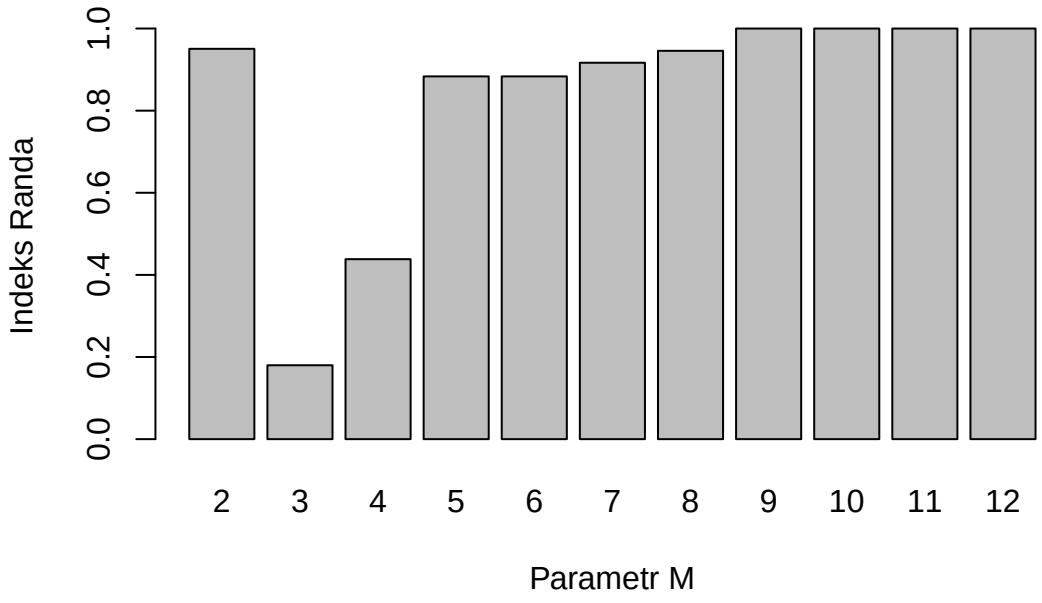

Figure 2: Zbiór 1 RN w iteracjach

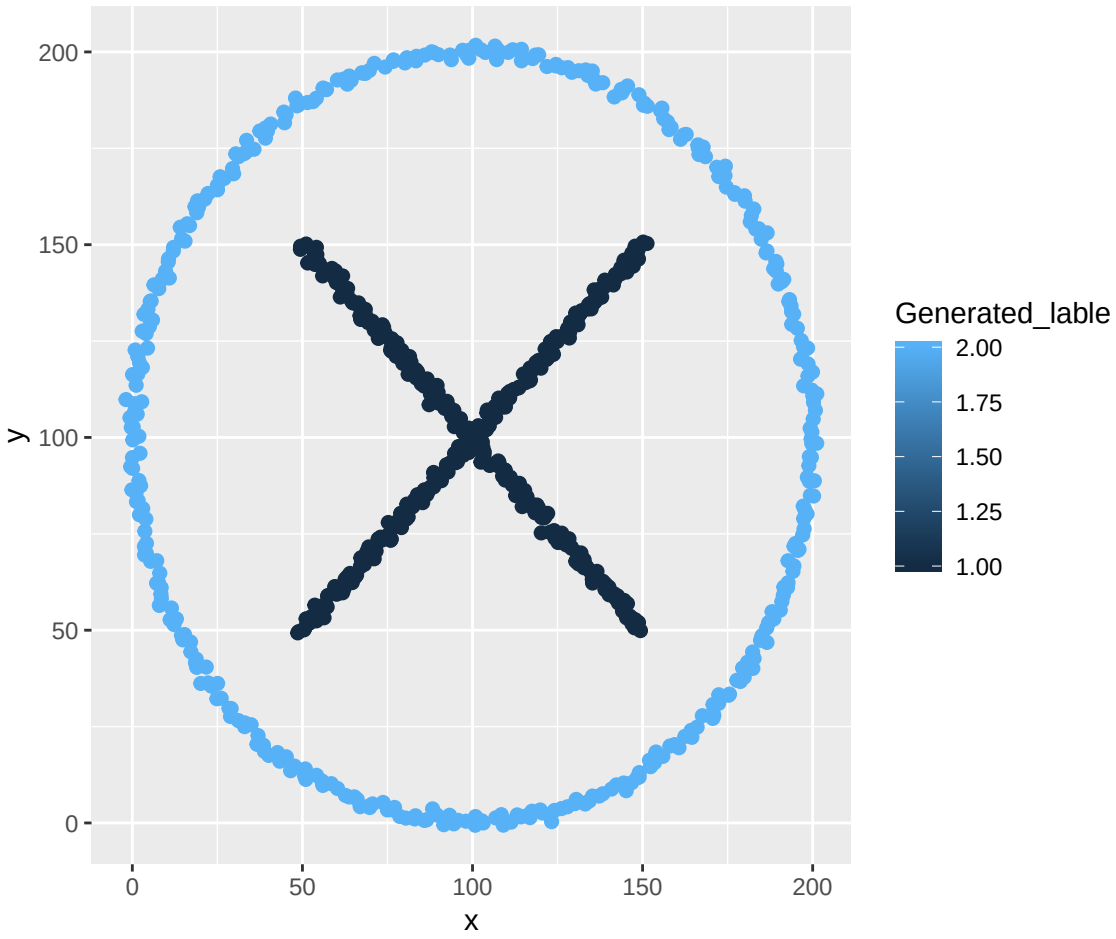

Figure 3: Zbiór 1 podział wygenerowany

Najlepszy podział :

```
## [1] "Skorygowany indeeks Randa  : 1"
## [1] "Indeks Fowlkesa-Mallowsa  : 1"
## [1] "Parametr M   : 9"
```

Jak widać w przypadku tego prostego zbioru podział jest zgodny, dla dość dużego zakresu **M**.

Zbiór 2

Drugi zbiór to dwie sfery o wspólnym środku jedna zawarta w drugiej zaburzonę o szum losowy. Sytuacja bardzo podobna do poprzedniaj ale teraz zbiór w 3D . Przedstawiony graficznie wygląda następująco :

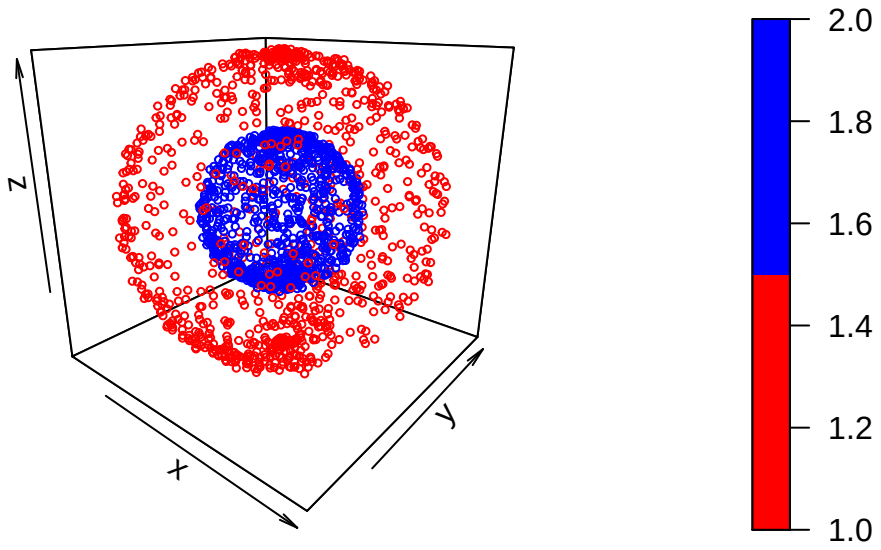

Figure 4: Zbiór 2 podział domyślny

Po wykorzystaniu algorytmu spektralnego do znalezienia podziałów :

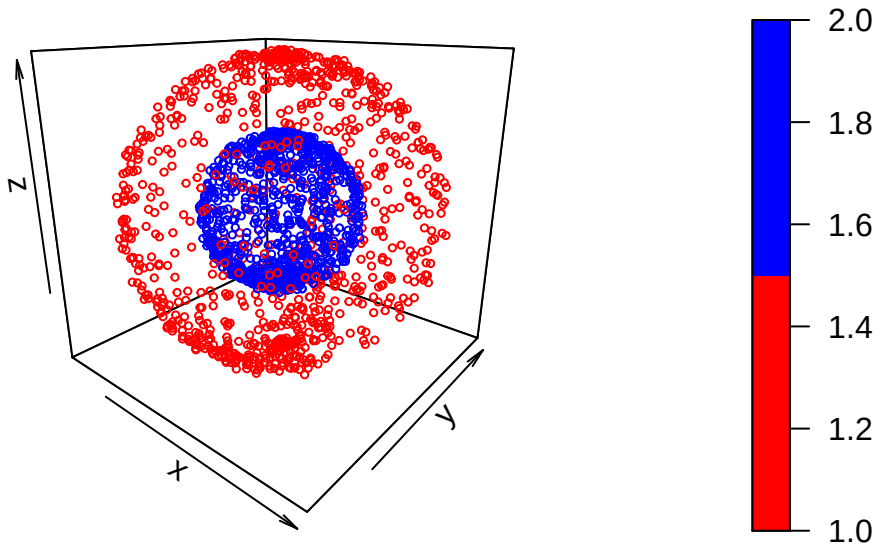

Figure 5: Zbiór 2 podział wygenerowany

```
## [1] "Skorygowany indeeks Randa  : 1"
## [1] "Indeks Fowlkesa-Mallowsa  : 1"
## [1] "Parametr M   : 4"
```

Jak widać również w tym wypadku algorytm radzi sobie bez zarzutu ale tak jak poprzednio nie dla karzdego **M**.

Zbiór 3

Ostatni z przestawionych przeze mnie zbiorów testowych ma na celu pokazanie problemów algorytmu który nie działa idealnie kiedy zbiory "przecinją sie". W tym celu rozważę zbiór w kształcie tak zwanych "kółek olimpijskich" tym razem bez szumu losowego . Przedstawione graficznie :

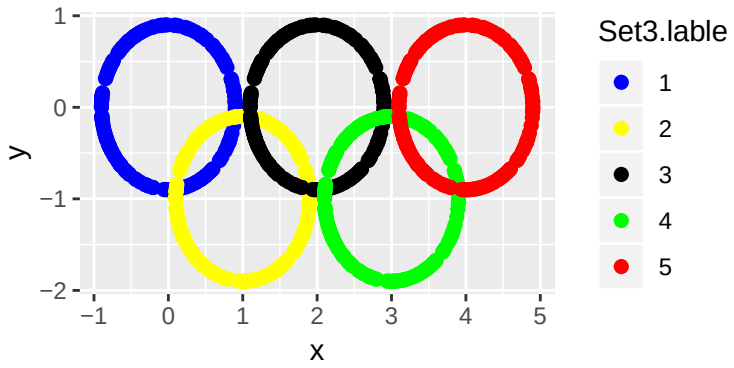

Figure 6: Zbiór 3 podział domyślny

W tym wypadku nie można po prostu przedstawić wyników działania algorytmu ponieważ potrafią się one diametralnie różnić w kolejnych itereciach pomimo niezmiennych parametrów **M** i **k**. Wyniki :

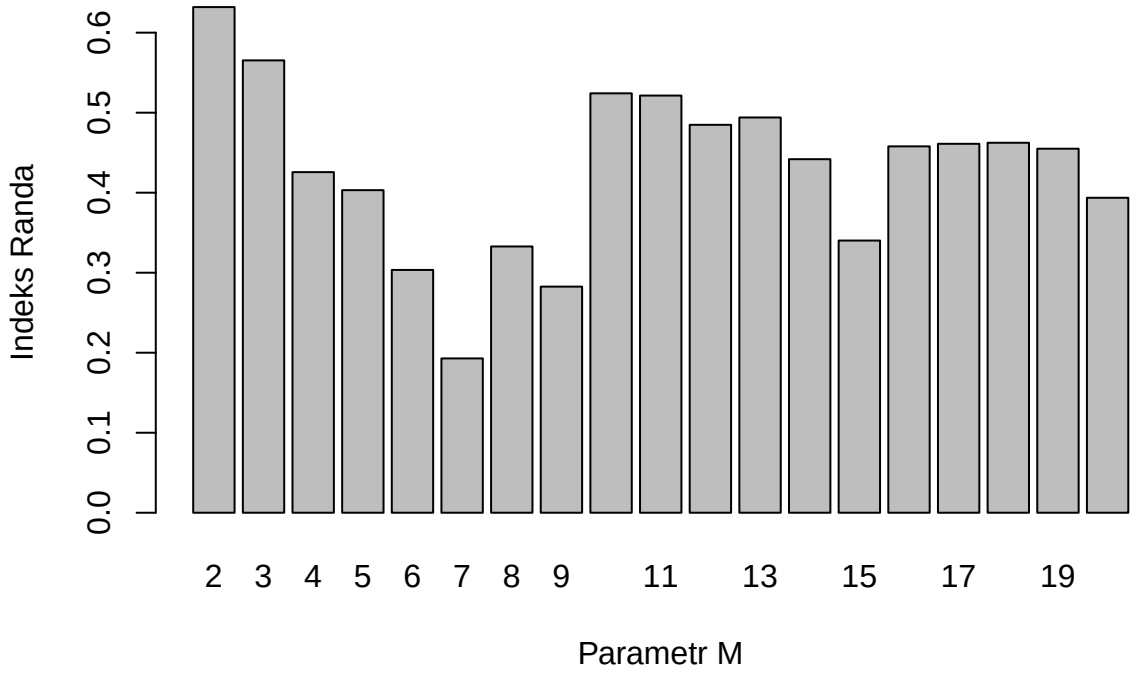

Figure 7: Parametr M dla zbioru 3

Jak widać najwyższy indeks występuje dla parametru $\mathbf{M} = 2$ więc z takim parametrem będe wywoływał algorytm w tym wypadku wywołam go 10 krotnie i policzę średnie indeksy : Wygenerowany podział :

```
## [1] "Średni skorygowany indeeks Randa  : 0.631978078865945"
## [1] "Średni indeks Fowlkesa-Mallowsa  : 0.705728688292604"
## [1] "Parametr M   : 2"
```

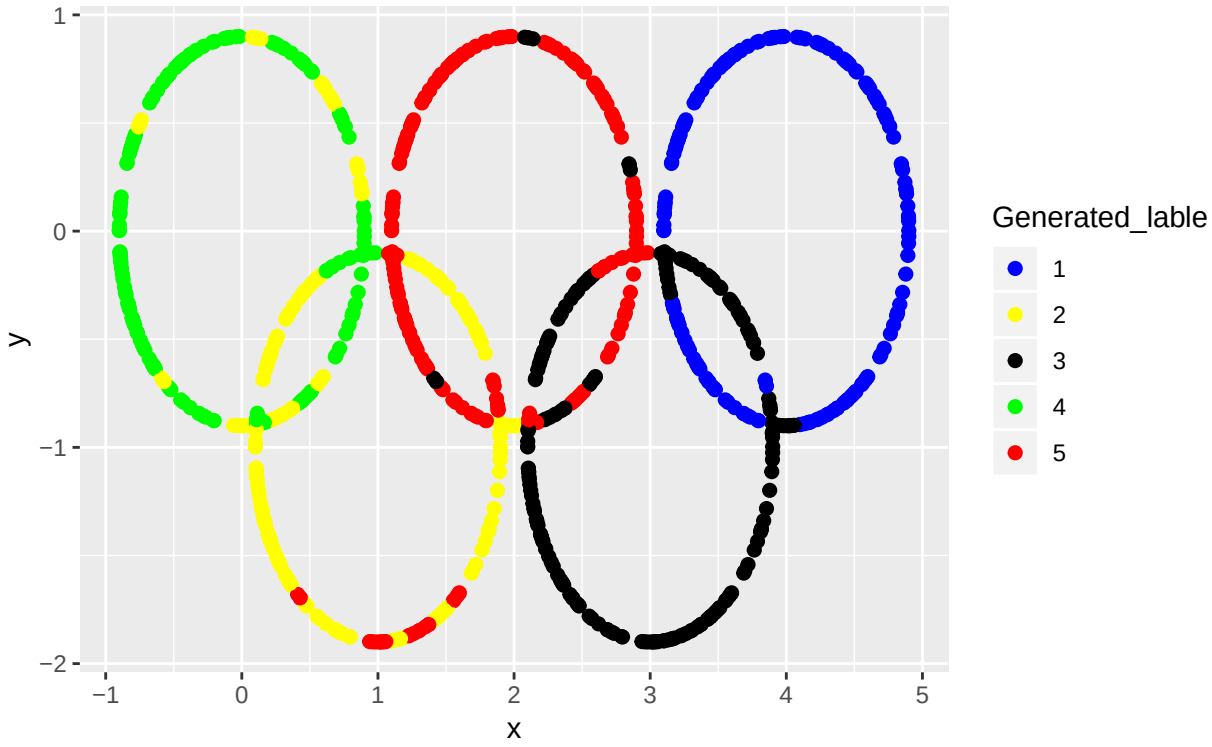

Figure 8: Zbiór 3 podziały wygenerowany

Jak widać w przypadku tego zbioru algorytm okazał się mało skuteczny jak mówiłem wcześniej wynika to z "przecinania się" .

Podsumowanie

Moja impementacja algorytmu spektralnego poradziła sobię dobrze ze zbiorami dostosowanymi do jej działana. 3 zbiór prezentuje problemy z algorytmem spektralnym który nie jest w nim wstanie dokonać odpowiedniego podziału, ponieważ najbliźsi sąsiedzi punktów nie są tu wystarczającą informacją. Pomimo tego uważam że moja implementacja tego algorytmu przeszła pomyślnie moje testy.