MWS Projekt

Znajdowanie punktów zmian w szeregach czasowych.

Tomasz Korzeniowski, 265753

23 stycznia 2018

1 Zadanie

Szereg czasowy to ciąg obserwacji pokazujący kształtowanie się badanego zjawiska w kolejnych okresach (sekundach, dniach, latach itp.). Badane zjawisko najczęściej ma swój typowy przebieg, lecz w wyniku wystąpienia zewnętrznego zdarzenia może zostać zaburzone. Wpływ zewnętrznego zdarzenia może objawiać się jako punkt zmiany. Punkt ten to moment, w którym następuje zmiana struktury obserwowanego szeregu czasowego. Oznacza to, że obserwacje poprzedzające punkt zmiany znaczenie różnią się od obserwacji następujących.

Będziemy rozważali N-elementowe wektory obserwacji. Przez x_i oznaczymy wartość obserwacji w chwili i-tej. Na potrzeby zadania zakładamy, że obserwacje są od siebie niezależne. Nie jest to zbyt silne założenie, ponieważ w każdym momencie wpływ zewnętrznego zjawiska może się pojawić lub zaniknąć.

Do reprezentacji punktów zmian wykorzystamy wektor $R = [r_i]_N$, w którym

$$r_i = \begin{cases} 1, & \text{gdy } x_i \text{ jest punktem zmiany} \\ 0, & \text{w przeciwnym przypaku} \end{cases}$$

Ponadto przyjmiemy, że $r_1 = r_N = 1$.

2 Metoda rozwiązania

Do wyznaczenia rozwiązania zostanie wykorzystany test Wilcoxona. Ten nieparametryczny test służy do sprawdzenia czy wartości próbek z dwóch niezależnych populacji są jednakowo duże.

Niech dwa segmenty $s_1 = \{x_{i^-+1}, \ldots, x_i\}$ o długości n_1 i $s_2 = \{x_{i+1}, \ldots x_{i^+}\}$ o długości n_2, gdzie i^- oraz i^+ oznaczają poprzedni i następny punkt zmiany, stanowią wybrane populacje.

Test polega na sprawdzeniu hipotez:

$$H_0 : \text{Populacje są sobie równe}$$
$$H_1 : \text{Populacje nie są sobie równe}$$

Test Wilcoxona składa się z następujących kroków:

1. Wszystkie obserwacje z globalnego segmentu $s = (s_1, s_2)$ uporządkuj w kolejności rosnącej pod względem wartości

2. Przypisz obserwacjom rangi będące kolejnymi liczbami naturalnymi. Jeśli wartości sąsiednich obserwacji są sobie równe, przypisaną im rangą jest średnia z rang jakie byłyby przypisane w przypadku gdyby wartości te były różne od siebie

3. Oblicz statystykę $U = \min(U_1, U_2)$:

$$U_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1, \quad gdzie\ R_1 - suma\ rang\ segmentu\ s_1$$

$$U_2 = n_1 n_2 + \frac{n_2(n_2+1)}{2} - R_2, \quad gdzie\ R_2 - suma\ rang\ segmentu\ s_2$$

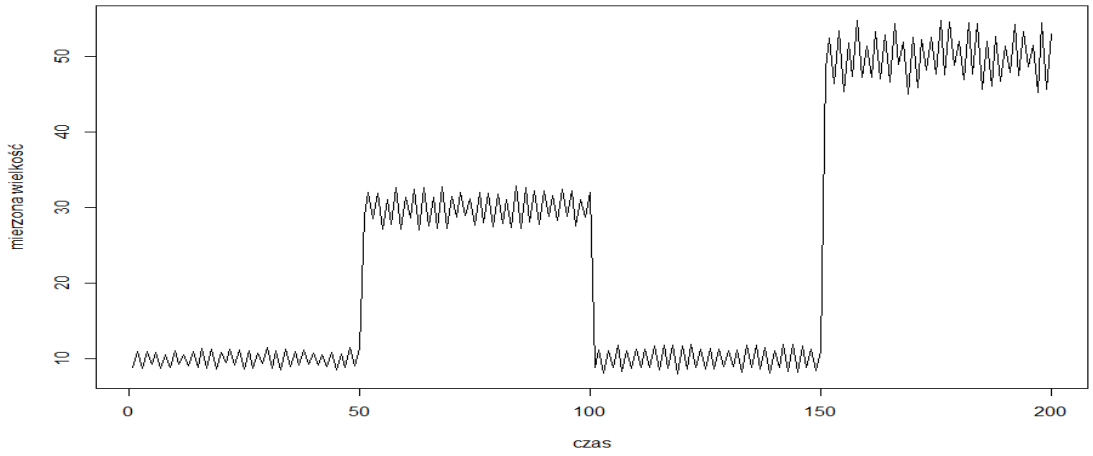
4. Wyznacz p-wartość dla zadanego poziomu istotności α

Jeżeli $p \leqslant \alpha$: odrzucamy H_0
Jeżeli $p > \alpha$: nie ma podstaw do odrzucenia H_0

Chcemy znaleźć wszystkie punkty zmian w zadanym przedziale czasowym. Tak jak autorzy [1], będziemy losowo wybierali chwilę i stanowiącą podział przedziału czasu na dwie populacje, a następnie wykonywali powyższy test. Powyższy podział szeregu na segmenty zapewnia, że punkt zmiany jest ostatnią obserwacją serii czasowej (podciągu szeregu czasowego o wspólnych właściwościach jak np. średnia).

3 Dane

Do weryfikacji poprawności działania testu wygenerujemy dane. Będą one miały przebieg okresowo stały z symulowanym szumem pomiarowym oraz dobrze widocznymi punktami zmian. Przykładowy przebieg prezentuje wykres 1.

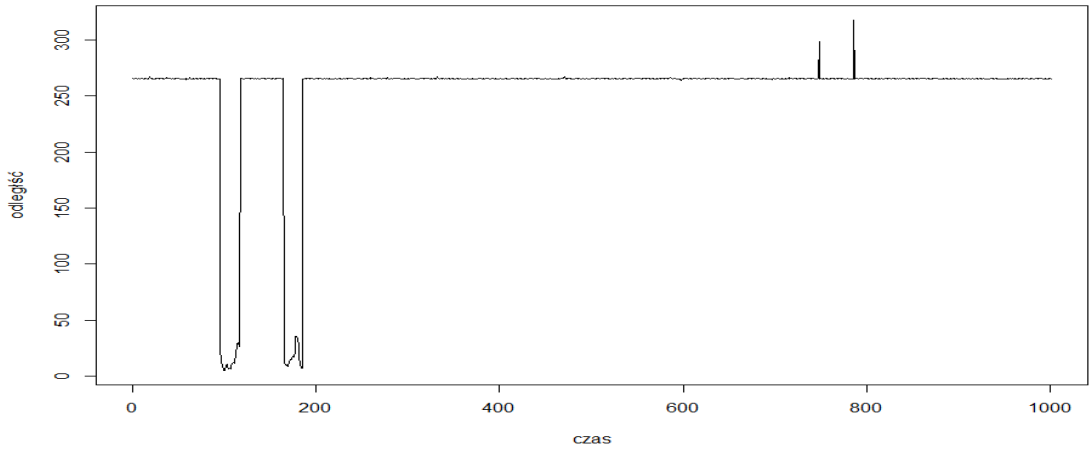


Wykres 1: Wygenerowany szereg czasowy.

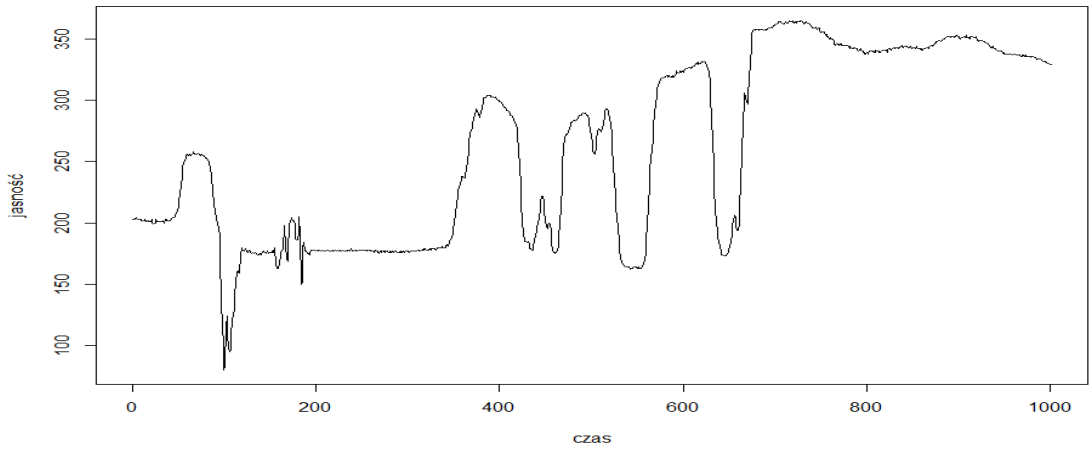
W kolejnym kroku skorzystamy z danych rzeczywistych. Dane, które będziemy rozważać, pochodzą z dwóch czujników zainstalowanych w pomieszczeniu biurowym. Pierwszy z nich sprawdza aktualną odległość od przeszkody (np. sufitu), a drugi zapisuje informację o jasności w pomieszczeniu. Częstość, z jaką czujniki zbierają odczyty nowych danych, to jedna sekunda. Przykładowe przebiegi odczytów przedstawiają wykresy 2 i 3.

Typowy przebieg odczytu odległości przypomina dane symulowane. Czujnik nie porusza się, więc jedyne zdarzenia jakie mogą się wydarzyć to przesunięcie przeszkody lub zasłonięcie czujnika. W przykładowym przebiegu widać także obserwacje odstające o wartościach większych niż typowa odległość od przeszkody.

Odczyt jasności jest zdecydowanie bardziej zawiły, lecz można w nim zaobserwować pewne chwile, gdzie nastąpiła nagła zmiana. Jednocześnie widać, że trudnej będzie zdecydować w którym miejscu powinien być znaleziony punkt zmiany.



Wykres 2: Przykładowy przebieg odczytu odległości.

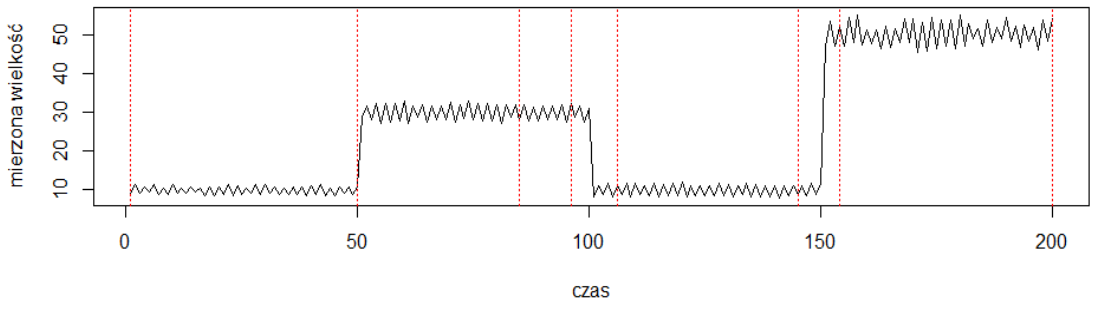


Wykres 3: Przykładowy przebieg odczytu jasności.

4 Wyniki

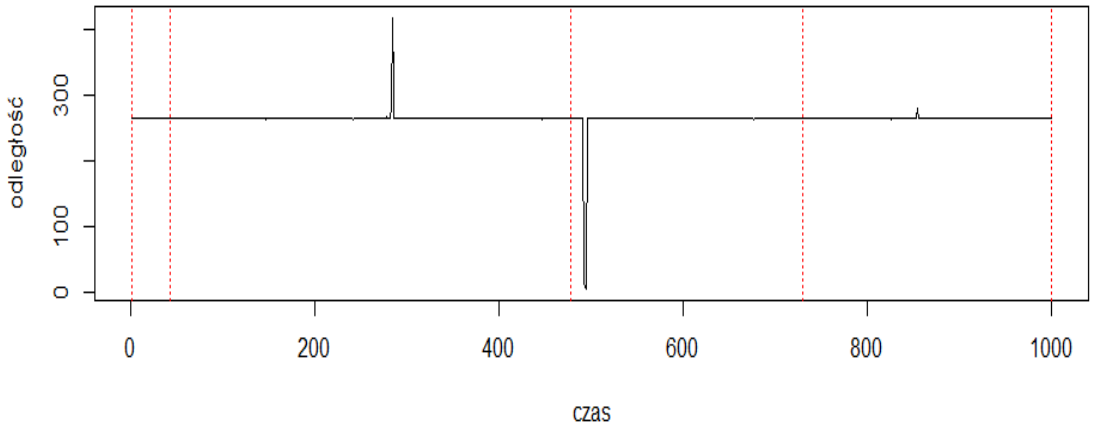
4.1 Losowe próbkowanie

Przykładowe wyniki dla wszystkich typów rozważanych danych prezentują wykresy 4 - 6. Przechodzimy po obserwacjach losowo, tzn. wybieramy dowolny punkt, dla którego sprawdzamy czy jest punktem zmiany. Jak można łatwo zauważyć, w przypadku danych wygenerowanych, jedynie jeden prawdziwy punkt zmiany został znaleziony poprawnie. Pozostałe dwa punkty zmian są wykryte niejawnie, tzn. algorytm wskazuje chwile bliskie prawdziwym punktom zmian. Ponadto istnieją także zupełnie fałszywe wskazania.



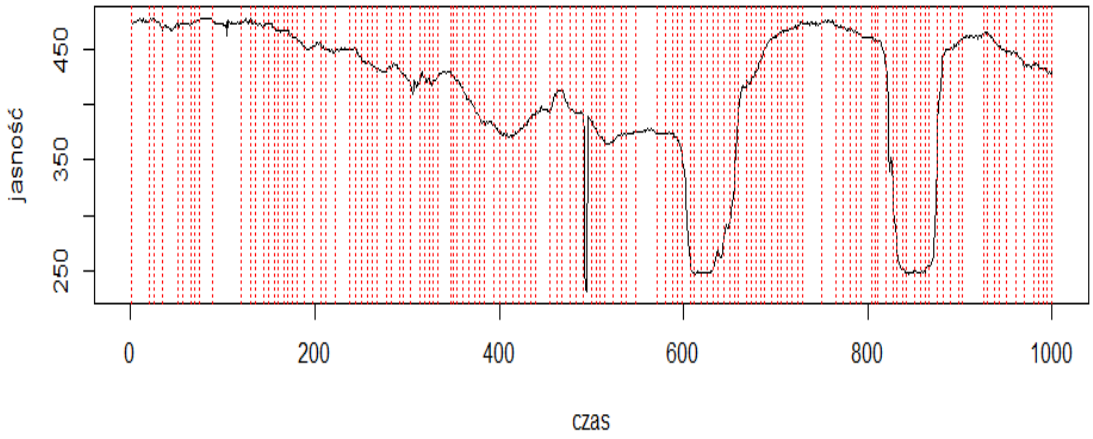
Wykres 4: Wygenerowany przebieg, losowa kolejność wyboru punktów.

W przypadku danych rzeczywistych, odczyt odległości (wykres 5) zawiera jedną obserwację odstającą i jedną serię czasową składającą się z kilku (około 5) obserwacji. Losowe próbkowanie nie wykrywa obserwacji odstającej, a w przypadku zbyt krótkiej serii czasowej wskazuje rozwiązanie dość bliskie, lecz mało dokładne. Błędne wskazania w chwili 42 i 729 wynikają najprawdopodobniej z lokalnych zaburzeń danych, niewidocznych w ogólnym przebiegu zjawiska.



Wykres 5: Odczyt odległości, losowa kolejność wyboru punktów.

Dla odczytu jasności, na wykresie 6, widać bardzo wiele znalezionych punktów zmian. Można stwierdzić, że algorytm nie nadaje się dla takich danych. Z drugiej strony, gdyby przyjrzeć się takiemu podziałowi w powiększeniu, byłoby widać, że obserwacje między kolejnymi dwoma punktami zmian mają bardzo zbliżone wartości. Jest to jednak zbyt duży podział, żeby uznać go za poprawny, gdyż jako obserwatorzy zjawiska spodziewamy się wykrycia „dużych" zaburzeń.

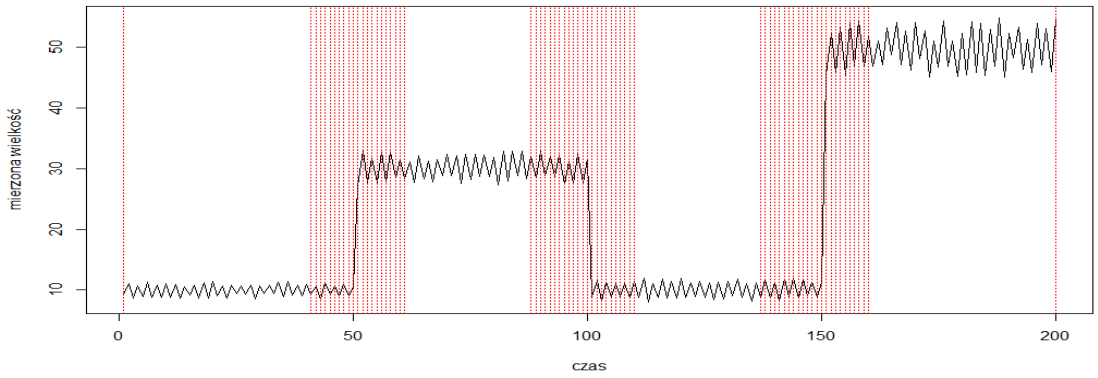


Wykres 6: Odczyt jasności, losowa kolejność wyboru punktów.

4.2 Rozwiązanie dokładne

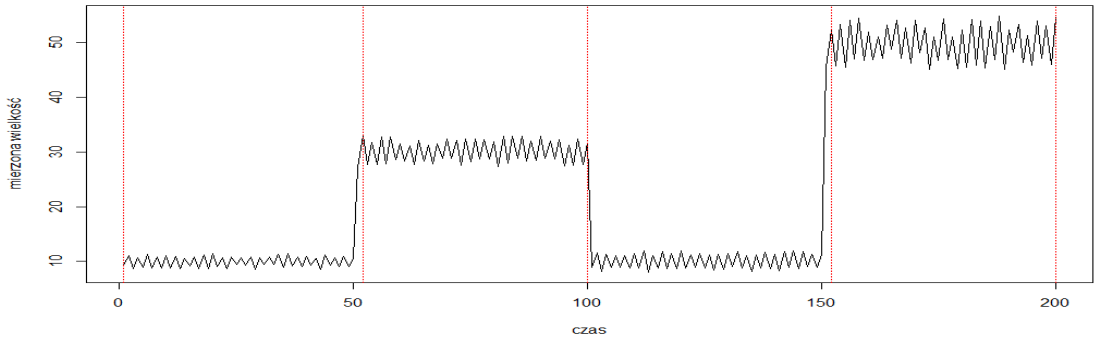
Na powyższych przykładach widać, że losowe próbkowanie nie sprawdza się najlepiej. Rozważamy modyfikację algorytmu polegającą na tym, że dane wejściowe będziemy traktować jakby pochodziły z pewnego strumienia, czyli napływały kolejno w pewnych odstępach czasu. Nowy algorytm będzie wykorzystywał pewne okno czasowe, w którym zostaną wyszukane punkty zmian.

Długość okna jest zmienna i wynika z położenia obserwacji. Jeśli badana obserwacja znajduje się między zaznaczonymi punktami zmian, okno czasowe przyjmuje długość tak wyznaczanego przedziału. Jeśli przedział ten jest dłuższy niż zadany parametr długości okna d, przedział należy ograniczyć by zawierał nie więcej niż d obserwacji przed oraz d obserwacji po badanej chwili czasowej. Przykład zastosowania tej modyfikacji przedstawia wykres 7.



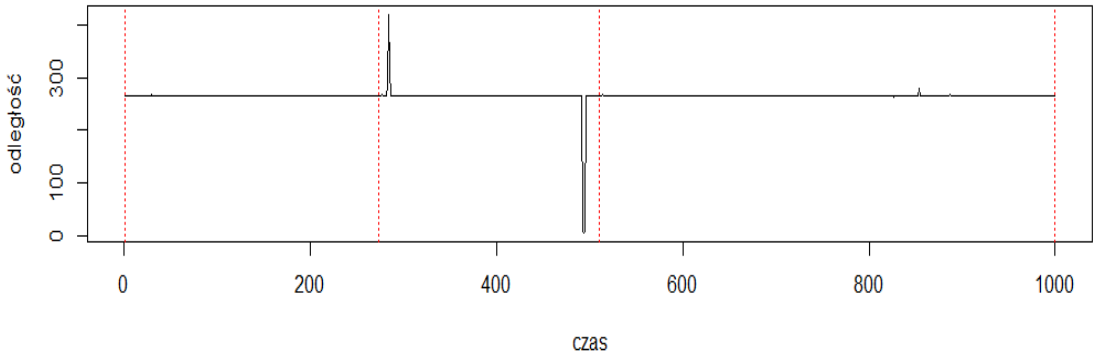
Wykres 7: Wygenerowany przebieg, badanie kolejnych obserwacji strumienia bez wygładzania.

Widać, że znalezione punkty zmian skupiają się wokół ich prawdziwych wystąpień, lecz w nadmiarowej liczbie zbliżonej do podwojonej długości okna d. Wyznaczając wartość środkową spośród każdej z grup punktów zmian, spodziewamy się poprawnego wykrycia prawdziwego punktu zmiany. Efekt ten prezentuje wykres 8. Należy zwrócić uwagę, że takie działanie jest poprawne, jeśli mamy do czynienia z danymi wygenerowanymi, czyli jednoznacznie widocznymi punktami zmian.



Wykres 8: Wygenerowany przebieg, badanie kolejnych obserwacji strumienia.

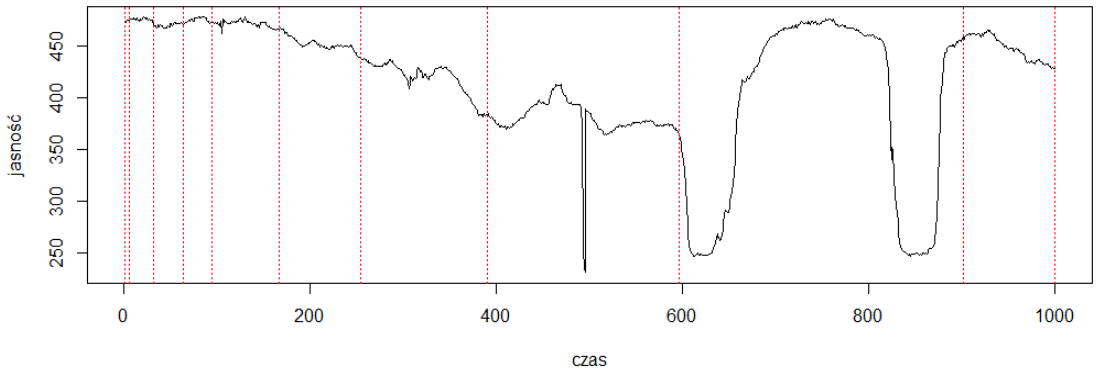
Zastosowanie metody dokładnej do danych rzeczywistych widać na wykresach 9 i 10.



Wykres 9: Odczyt odległości, badanie kolejnych obserwacji strumienia.

W tym przypadku obserwacja odstająca w odczycie odległości nie jest pomijana, lecz ponownie znaleziona chwila rozmija się z rzeczywistym wystąpieniem zdarzenia. To samo zachodzi dla bardzo krótkiej serii czasowej. Mimo że punkty zmian nie są znalezione dokładnie tam gdzie powinny, widać, że nie ma nadmiarowych wskazań, co świadczy o poprawie w stosunku do losowego próbkowania.

Odczyt jasności jest podzielony znacznie bardziej gruboziarniście niż w losowym próbkowaniu. Cały czas można wskazać miejsca, gdzie rzeczywiste punkty zmian nie zostały znalezione, lecz wyznaczony podział bardziej oddaje spodziewany wynik (zaznaczenie głównych miejsc, gdzie nastąpiło zdarzenie).



Wykres 10: Odczyt jasności, badanie kolejnych obserwacji strumienia.

Na potrzeby testów został przyjęty poziom istotności $\alpha = 0.01$. Jeśli poziom istotności zostanie zmniejszony to w przypadku losowego próbkowania otrzymamy więcej fałszywych wskazań. Dla metody dokładnej, zmniejszenie poziomu istotności powoduje niewielkie przesunięcie dotychczas znalezionych (np. wskazanie sąsiedniej obserwacji niż przy wyższym poziomie istotności) lub dodatkowe punkty zmian.

4.3 Wnioski

Zaprezentowane sposoby znajdowania punktów zmian w szeregach czasowych nie są idealne. Największym problemem jest wykrywanie obserwacji odstających. Jeśli przebieg jest typowy (wykres 1) łatwo znaleźć poprawne rozwiązanie, lecz w przypadku danych rzeczywistych należałoby opracować dodatkowe kryteria oceny czy dana obserwacja jest punktem zmiany. Należałoby także dostosować długość okna do charakterystyki danych.

5 Implementacja

W celu rozwiązania zadania jedną z dwóch, zaprezentowanych wcześniej, metod należy wywołać funkcję *getChangepoints*. Jej parametrami wejściowymi jest zbiór danych, w którym będą poszukiwane punkty zmian, poziom istotności α oraz flaga decydująca o metodzie rozwiązania. Jeśli flaga będzie ustawiona na $FALSE$ (domyślnie) zostanie wybrana metoda dokładna, w przeciwnym przypadku rozwiązanie zostanie wyznaczone metodą losowego próbkowania.

Literatura

[1] F. Harle, F. Chatelain, C. Gouy-Pailler, S. Achard *Rank-based multiple change-point detection in multivariate time series.* Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European. IEEE, p. 1337-1341, 2014

Dodatek A Implementacja rozwiązań

```
getChangepoints <- function(data, alfa=0.01, random = FALSE){
  if(nrow(data)<ncol(data)){ # kazda kolumna odpowiada wymiarowi danych
    data<-t(data)
  }

  N <- nrow(data) # liczba obserwacji
  K <- ncol(data) # liczba wymiarow
  R <- matrix(0, N, K) # znalezione punkty zmian
  R[c(1,N), c(1:K)] <- 1

  P <- matrix(0, N, K) # p-wartosci dla kazdego testowanego podzialu

  if(!random){ # przechodzenie po danych po kolei z zadana dlugoscia okna
    dlug_okna = 20;
    for (i in 2:(N-1)) {  # losowy punkt
      if((i!=1) & (i!=N) ){

        for (j in 1:K){ # kazdy wymiar niezaleznie
          f <- which(R[,j] > 0) # znajdowanie podzialu na probki
          i_ <- f[max(which(f < i))] # indeks poprzedniego punktu zmiany
          ip <- f[min(which(f > i))] # indeks nastepnego punktu zmiany

          if (ip > i + dlug_okna) ip <- i + dlug_okna;
          i_ <- i - dlug_okna;

          if(i_<1) i_<-1
          if(ip>N) ip<-N

          s1 <- data[(i_+1):i, j] # podziel probki
          s2 <- data[(i+1):ip, j]

          # test Wilcoxona
          wt<-wilcox.test(as.matrix(s1), as.matrix(s2), exact = FALSE, correct =
              FALSE)
          if(!is.finite(wt$p.value)){
            P[i,j] <- pwilcox(wt$statistic, ncol(as.matrix(s1)), ncol(as.matrix(
                s2)))
          }else{
            P[i,j] <- wt$p.value
          }

          if(P[i,j]<=alfa){  # odrzucamy hipoteze zerowa
            R[i,j]<-1
          }else{
            R[i,j]<-0 # nie ma podstaw do odrzucenia H0
          }

        }

      }

    }

    R2 <- matrix(0, N, K) # odfiltrowanie niepoprawnych punktow zmian
    R2[c(1,N), c(1:K)] <- 1
    for(j in 1:K){ # dla kazdego wymiaru
      i <- 2
      while(i<N-1){
        if(R[i, j]==1){
          k <- i+1
          while(R[k, j]!=0){k <- k+1}
```

```
          rr <- round((i+k)/2)
          R2[rr, j] <- 1  # wybor srodkowej obserwacji jako punkt zmiany
          i <- i+k/2
        }else{
          i <- i+1
        }

      }
    }
    R2
  }
  else
  { # przechodzenie po danych losowo
    for (i in sample.int(N)){  # losowy punkt
      if((i!=1) & (i!=N) ){

        for (j in 1:K){ # kazdy wymiar niezaleznie

          i_ <- i-1 # indeks poprzedniego punktu zmiany
          ip <- i+1 # indeks nastepnego punktu zmiany

          f <- which(R[,j] > 0) # znajdowanie podzialu na probki
          i_ <- f[max(which(f < i))]
          ip <- f[min(which(f > i))]

          if(i_<1)i_<-1
          if(ip>N)ip<-N

          s1 <- data[(i_+1):i, j] # podziel probki
          s2 <- data[(i+1):ip, j]

          # test Wilcoxona
          wt<-wilcox.test(as.matrix(s1), as.matrix(s2), exact = FALSE, correct =
              FALSE)
          if(!is.finite(wt$p.value)){
            P[i,j] <- pwilcox(wt$statistic, ncol(as.matrix(s1)), ncol(as.matrix(
                s2)))
          }else{
            P[i,j] <- wt$p.value
          }

          if(P[i,j]<=alfa){  # odrzucamy hipoteze zerowa
            R[i,j]<-1
          }else{
            R[i,j]<-0 # nie ma podstaw do odrzucenia H0
          }

        }

      }

    }
    R
  }
}
```

A.1 Przykładowe wywołanie

```
# wyniki dla danych wygenerowanych
N <- 200
X <- matrix(0, 200, 1)

for(i in 1:50){
  X[i]<-10 +((-1)^i)*runif(1, .5, 1.5)
  X[50+i]<-30 +((-1)^i)*runif(1, 1.0, 3.0)
  X[100+i]<-10 +((-1)^i)*runif(1, 1.0, 2.0)
  X[150+i]<-50 +((-1)^i)*runif(1, 1.0, 5.0)
}

r <- getChangepoints(X, random = TRUE)

plot(c(1:N), X, type = 'l', xlab='czas', ylab='mierzona wielkosc')
abline(v=c(which(r==1)), col="red", lty=3, lwd=1)


# wyniki dla danych rzeczywistych
data <- read.table('dane.txt')
colnames(data) <- c('odleglosc', 'jasnosc')

# przykladowy przebieg danych
plot(c(1:1001), data[6900:7900,1], type='l', xlab='czas', ylab='odleglosc')
plot(c(1:1001), data[6900:7900,2], type='l', xlab='czas', ylab='jasnosc')

data<-data[1:1000, ] # demonstracja wynikow na mniejszym przedziale

R <- getChangepoints(data, alfa = 0.05) # wyniki


N <- nrow(data) # liczba obserwacji
plot(c(1:N), data[, 1], type = 'l', xlab='czas', ylab='odleglosc')
abline(v=c(which(R[,1]==1)), col="red", lty=3, lwd=1)

plot(c(1:N), data[,2], type = 'l', xlab='czas', ylab='jasnosc')
abline(v=c(which(R[,2]==1)), col="red", lty=3, lwd=1)
```