# Izločanje očesnih artefaktov z uporabo postopka analize neodvisnih komponent (ANK)

*Marcel Polanc (63170240)*

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko

Datum: 17.01.2021

## 1 Uvod

Za temo sem si izbral nalogo imenovano Izločanje očesnih artefaktov z uporabo postopka analize neodvisnih komponent (ANK). V signalih EEG lahko pride do šuma, to pa zaradi različnih dejavnikov. Eden izmed dejavnikov je lahko tudi utripanje oči in čiščenje teh šumov je ključnega pomena. V poročilu bom najprej opisal metode seminarske naloge torej teoretično ozadje. Nato bom opisal moje rezultate, do katerih sem prišel pri reševanju te naloge (s pomočjo programske opreme MatLab). Na koncu v diskusiji pa bom povzel moje poglede in moja spoznanja ob izdelavi naloge.

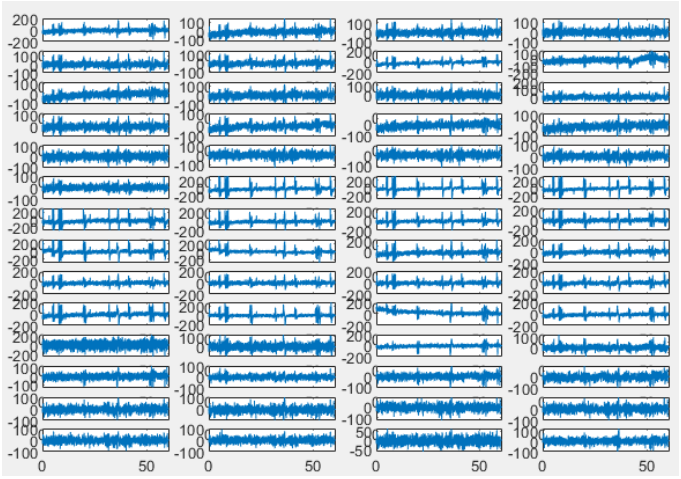
## 2 Metode

Pri seminarski nalogi se pojavi metoda oziroma postopek imenovan analiza neodvisnih komponent (ANK). Pri postopku ANK gre za numerično metodo, ki iz izmerjene linearne mešanice signalov na podlagi zgolj njihovih statičnih lastnosti rekonstruira izvirne oziroma statično neodvisne signale.

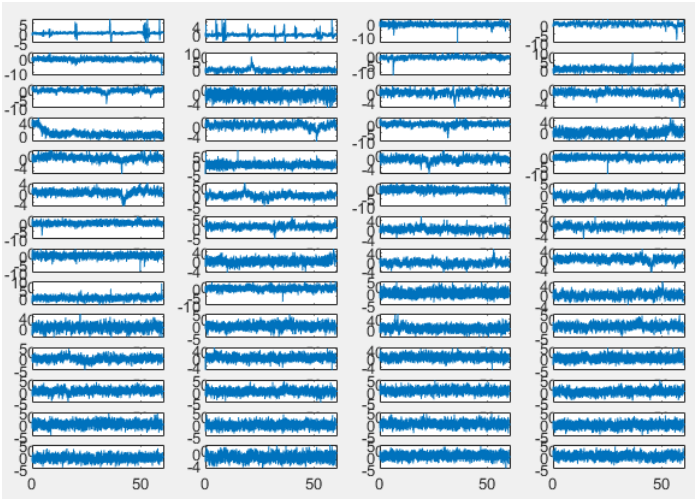
Pri ANK gre za prostorski filter, prostorski filtri pa transponirajo X(n) signal, ki je N-kanalni v signal Y(n). Kateri ima n = 1,…,N. Pri tem pa dobimo enačbo Y(n) = W * X(n). Pri enačbi pa gre za transformacijsko matriko imenovano W, katera slika prostor signalov v prostor komponent. Naslednji korak pa je da matriko W transformiramo v matriko W'.

## 3 Rezultati

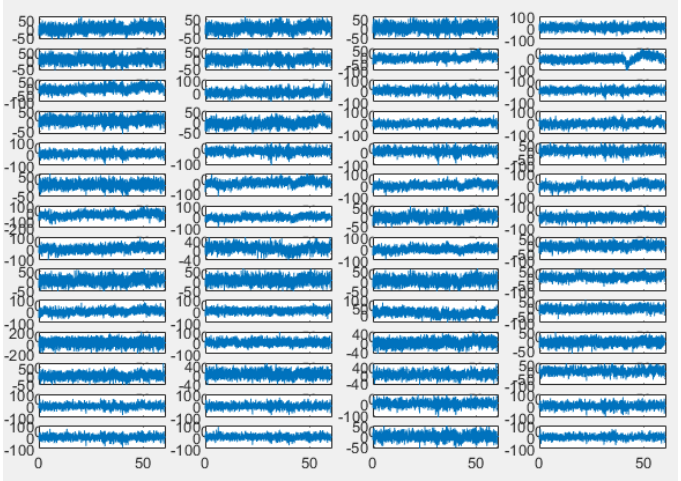
Pri moji implementaciji seminarske naloge, sem najprej s podatkovne baze EEGMMI DS prenesel nekaj posnetkov elektrod. Nato sem v programski kodi z ukazom rdsamp, s prenesenega posnetka prebral signal v tri spremenljivke eeg, freq in tm. Potem sem izvedel ukaz fastica nad eeg spremenljivko in dobil komponente icasig, A in W. Kjer so icasig neodvisne komponente, A je mešalna matrika in imamo še W. Nad mešalno matriko A, ki je približek W sem potem izvedel po stolpcih filtriranje in sicer sem gledal vrstice 22, 23 in 24, ki predstavljajo očesne komponente. Če je presegla vrednost nek treshold v mojem primeru 8 sem potem moral odstraniti ujemajoč stolpec v matriki A in ujemajočo vrstico v matiki icasig. Na način sem potem odstranil artefakte, naslednji korak pa je bil da sem množil novo pridobljeno matriko A in icasig in to je predstavljalo moj končen rezultat seminarja. Že prej sem sproti prikazoval grafe in sicer sem najprej prikazal osnovni graf signala. Osnovni graf signala je prikazan na spodnji sliki.



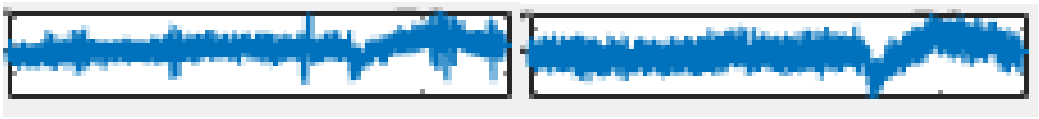
Potem, ko sem izvedel izločil posamezne komponente sem prikazal graf signalov v prostoru komponent. To prikazuje spodnja slika.



In še po izvedbi metode ANK sem prikazal graf signalov oziroma sem prikazal graf korigiranih signalov. To pa prikazuje spodnja slika. In to je bil tudi rezultat mojega seminarja. Lepo je razvidno, kako so se s signalov izločili utripi oči, ker ni več znatnih hitrih visokih sprememb oziroma vrednosti znotraj posameznih signalov.



Če podrobneje pogledamo en izmed signalov za primer sem vzel signal ki je osmi po vrsti, je lepo razvidno, kako se porežejo očesni artefakti oziroma utripi oči. To razliko originalnega signala in korigiranega signala številke osem prikazuje spodnja slika.



# 4 Diskusija

Pri izdelavi te seminarske naloge sem se naučil veliko novih zadev glede filtriranja signalov. Ob sami izdelavi sem imel kar veliko dela s prej omenjenim delom kode, kjer izločim določene vrednosti s signala, katere so izven območja +/- treshold. Namreč pri implementaciji sem to izvedel tako, da sem naredil začasen vektor, v katerega sem zaradi lažje izvedbe shranjeval najprej vse instance vrstic, katere so primerne in ne presegajo tresholda. Zaradi tega pa sem potem potreboval narediti še en vektor, ki je imel vse vrednosti od 1 do dolžine stolpcev metrike A. Potem sem preprosto odštel ozirom s tega vektorja, ki ima vse vrednosti odstranil prejšnji začasni vektor, ki vsebuje dobre instance in na tak način sem prišel do vektorja, ki vsebuje vrednosti katere je potrebno odstraniti. Nato sem nadaljeval po postopku ki je opisan zgoraj. Prav tako sem imel pri implementaciji nekakšne težave predvidevam da s pomnilnikom oziroma »slabim« računalnikom, ker je izvedba moje kode trajala kar nekaj časa in včasih se sploh ni izvedla, zato sem se odločil, da sem namesto 64 (oz. 65) signalov torej vseh signalov ki se pojavijo uporabil le 56 signalov. In ko sem izvedel to zadevo je potem bila koda vsakič izvedljiva in lepo sem prišel do rezultatov in izločenih artefaktov kot zahteva naloga.