LAMPIRAN

List Program

```
Proyeksi dasar algoritma pencarian pada 2 menunjukkan sinyal suara tetapi hanya satu sinyal yang di ekstrak
clear;
close all;
clc;
% Hanya untuk mendengarkan 1 sinyal
listen=1; % set to 1 if have audio.
% Menetapkan nomer acak
seed=99;
rand('seed',seed);
randn('seed',seed);
% M = Jumlah sumber sinyal dan sinyal campuran
M = 2;
% [1e4] N = Jumlah titik data sinyal
% N = 1e4;
% Mengambil data, masing-masing M = 2 baris berisi sinyal sumber yang berbeda.
% Setiap baris mempunyai N kolom atau nilai dari sinyal
% Atur varians setiap sumber untuk campuran.
% Mulai memasukkan input
[spe fsspe jbit]=wavread('Speech');
spe1=spe(1000:end);
pjg1=length(spe1);
[noi1 fsnoi1 jbit]=wavread('noise1');
noi11=noi1(1000:end);
pjg2=length(noi11);
pendek=min(pjg1,pjg2);
N=pendek;
```

```
spe2=spe1(1:pendek);
noi2=noi11(1:pendek);
s1=spe2';
s2=noi2';
% Kombinasikan variabel vektor s kedalam sumber
s=[s11; s2];
% Percampuran matriks
A=randn(M,M)';
% Mendengarkan sumber sinyal
% Frekuensi sampling = 10000.
Fs=8000;
% Gambarkan setiap sumber sinyal
figure(3);
hist(s(1,:),50);
drawnow;
figure(4);
hist(s(2,:),50);
drawnow;
% Membuat campuran sinyal dari sumber sinyal
x=A*s;
% Dengarkan hasil pencampuran sinyal
if listen
   soundsc(x(1,:),Fs);
   soundsc(x(2,:),Fs);
end;
% Membuat sinyal campuran menggunakan SVD.
[U D V]=svd(x',0);
% Mengganti nilai vektor x dengan memasukan nilai x yang baru .
z=U;
```

```
% Mengubah nilai kombinasi dari eigenvektor yang ada  ...
z=z./repmat(std(z,1),N,1);
z=z';
% Pemisahan sinyal menginisialisasi vektor – vektor acak
w = randn(1,M)';
% Dengan menggunakan satuan panjang w
w=w/norm(w);
% Menginisialisasi setiap sinyal sumber.
y = w'*z;
% Membuat perkiraan hubungan korelasi antara sumbu sinyal sumber (s)
fprintf('Initial correlations of source and extracted signals\n');
%rinitial=abs(r(M+1:2*M,1:M))
r1=corrcoef([y; s1]');
r2=corrcoef([y; s2]');
rinitial=abs([r1(1,2) r2(1,2)])
maxiter=100; % [100] Maximum number of iterations.
eta=2e-2; % [1e-2 /2] Step size for gradient ascent.
% Membuat sebuah array hs untuk menyimpan nilai-nilai gradien fungsi dan ukuran.
Ks=zeros(maxiter,1);
gs=zeros(maxiter,1);
%  Awal permulaan kenaikan nilai K
% Memasukan nilai terbaik untuk beban vektor
wopt=[-0.6125 0.7904];
for iter=1:maxiter
   % Memperkirakan nilai sumber sinyal y
   y = w'*z;
   % Estimasi nilai kurtosis.
   K = mean(y.^4)-3;
   % Menemukan nilai gradient K dan w
```

```
    y3=y.^3;
    yy3 =repmat(y3,2,1);
    g=mean((z.*yy3)')';
    % Memasukan nilai w untuk meningkatkan nilai K
    w = w + eta*g;
    % Menetapkan panjang nilai w
    w = w/norm(w);
    % Menunjukkan skala dan sudut antara wopt dan gradient K
    Ks(iter)=K;
    gs(iter)=subspace(g,wopt');
end;
% Mencatat perubahan plot dan sudut gradien wopt selama pengoptimasian
figure(1);plot(Ks,'k');
title('Function values - Kurtosis');
xlabel('Iteration');ylabel('K(y)');
figure(2);plot(gs,'k');
title('Angle \alpha Between Gradient g and Final Weight Vector w');
xlabel('Iteration');ylabel('\alpha');
% Menampilkan nilai korelasi akhir
r=corrcoef([y;s]');
fprintf('Final correlations between source and extracted signals ...\n');
r1=corrcoef([y;s1]');
r2=corrcoef([y;s2]');
rfinal=abs([r1(1,2) r2(1,2)])
% mendengarkan sinyal diekstrak
if listen
    soundsc(y,Fs);
end;
```