

第3章 音声・画像の信号分離

3.1 実験目的

電子情報処理分野において、音声、画像、通信データ等の信号処理が主な課題としてしばしば取り上げられる。本実験の目的は、近年発展の著しいブラインド信号処理(Blind Signal Processing)分野の応用として信号分離(Source Separation)問題を取り上げ、MATLAB のプログラミングを通して実際に信号分離を行うことである。また、音声・画像データ等の取り扱い方を習得する。更に混合信号分離問題への数式化を行い、その数理モデルの解析および計算アルゴリズムの設計方法について学ぶ。

3.2 実験原理

3.2.1 混合信号の分離方法

1. 信号分離問題の背景

電子情報工学などの分野では、複数の信号成分が互いに混じり合っている混合信号から、目的とする信号成分のみを取り出すという問題がしばしば現れる。例えば、何人かが同時に発話した音声をマイク（センサー）によって収録した混合信号から、各話者に対応する音声信号に分離する問題がある。またノイズに埋もれた画像から元画像に復元する問題もある。これらの問題は、近年、ブラインドセパレーション(Blind Source Separation)の枠組みでその解決が試みられている。また、90 年代の中期頃に独立成分解析(Independent Component Analysis)が提唱され、混合信号分離問題の有力な一つの手法として注目されている。

2. 信号分離問題の定式化

信号分離の目標は、混合された信号を個々の原信号（または信号源）に分離することである。この信号分離の問題を効率よく検討するためには、まず数式で問題を記述すること（すなわち問題の定式化）が必要である。今、2 人同時に発話された音声信号の分離問題について考えてみよう。ある場所で A さんと B さんが同時に喋っている。ここで、A さんと B さんの声を信号源としてそれぞれ s_1 と s_2 とする。その発話された音声信号を記録するために、二つのマイクロフォン 1 と 2 が設置されていると仮定し、各々のマイクロフォンによって記録された信号を x_1 と x_2 とする。また、信号源から各マイクロフォンまでの伝達特性は、係数 a_{11} 、 a_{12} 、 a_{21} 、 a_{22} のみを用いて簡略化されるとすると、マイクロフォン 1 で記録された混合信号 x_1 は、式(1)で表現することができる。

$$x_1(t) = a_{11}s_1(t) + a_{12}s_2(t) \quad t = 1, 2, \dots \quad (1)$$

上式において、係数 a_{11} と a_{12} は信号源 s_1 と s_2 からマイクロフォン 1 までの物理距離を示していると考えればよい。すなわち、信号源とマイクロフォンの距離が近ければ、係数の値が大きくなり（つまり、音声信号の振幅が増加する）、信号源がマイクロフォン信号 x_1 を占有している割合も大きくなる。ここで、 $t=1, 2, \dots$ は離散時間とする。

同様にして、マイクロフォン 2 で記録された混合信号 x_2 は式(2)で表現できる。

$$x_2(t) = a_{21}s_1(t) + a_{22}s_2(t) \quad (2)$$

以上、音声分離の問題を簡単な数式(1)と(2)で記述することができた。つまり、今我々が解こうとしている信号源の分離問題は、上述した問題の定式化により、式(1)と(2)の s_1 と s_2 の解を求める、という数式上の問題に帰着することができる。しかし、実際には係数 a_{11} 、 a_{12} 、 a_{21} 、 a_{22} は未知であり、式(1)と式(2)のみを利用して信号源 s_1 と s_2 を求めることは不可能である。そのため、何らかの方法を用いて s_1 と s_2 の（少なくとも）近似解を求める必要がある。これについては次節に詳しく説明する。

簡潔のため、式(1)と式(2)をまとめて次式のように行列形式で書き直す。

$$\mathbf{x}(t) = A\mathbf{s}(t) \quad (3)$$

ここで、マイクロフォン信号は縦ベクトル $\mathbf{x}(t) = [x_1(t) \ x_2(t)]^T$ (T は転置の記号である。)、

信号源は $\mathbf{s}(t) = [s_1(t) \ s_2(t)]^T$ 、また、伝達特性を表す係数行列は $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ となる。

例題 1： ホームページ(<http://www.sit.ac.jp/user/cao/Teaching/Teaching.htm> 第6章 音声データ)から「moshimoshi」と「birdchirps」という音声信号（各々信号源信号として用いる）をダウンロードし、ハードディスクDにあるフォルダ **MYWORK** に保存する。次のプログラムを実行し、実際に元の音声と混合された音声信号を視聴して確かめよう。また、以下のプログラムでは、二つのマイクロフォン信号にあたる混合信号が保存される。

```

                                プログラム   samp1.m
% Original speech signals and mixed signals
% filename: samp5_1.m
% Jianting Cao, Aug. 10, 2003

% Workspace 変数をすべて削除
clear all

% 音声信号の読み込み、信号源の生成
load birdchirps1;
load mosimosil;
s=[birdchirps1;mosimosil];

% 音声信号を試聴する
clc;input('鳥の鳴き声 ... ')
sound(birdchirps1);
clc;input('もしもし ... ')
sound(mosimosil);

% randn() を用いて混合係数を生成
A=randn(2,2);

```

```

% 混合信号の生成
x=A*s;

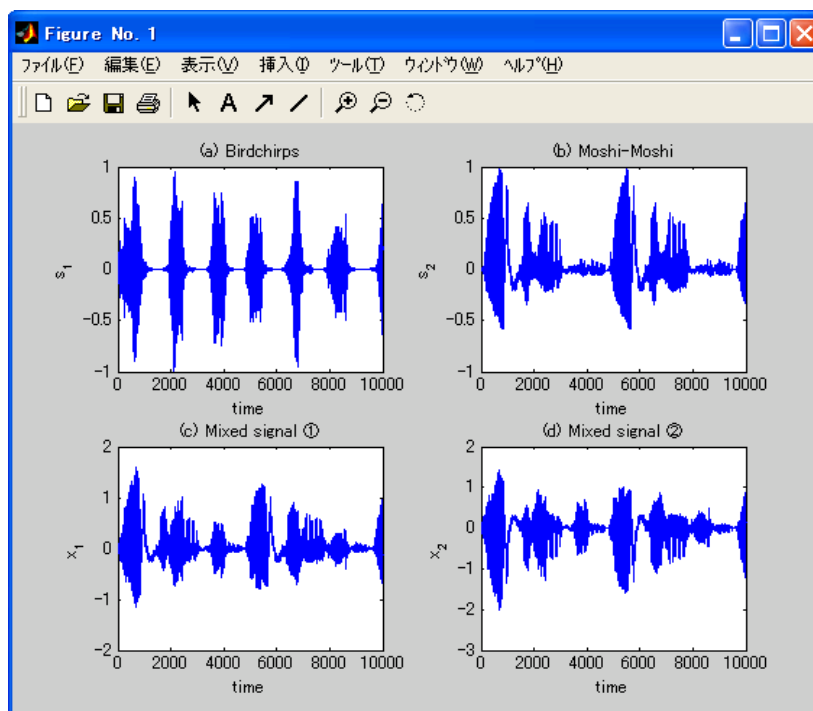
% 混合信号を試聴する
clc;input('混合信号① ... ');
sound(x(1,:));
clc;input('混合信号② ... ');
sound(x(2,:));

% 混合信号の保存
save x x A

% 信号波形の表示
subplot(2,2,1);plot(s(1,:));
xlabel('time');ylabel('s_{1}');title('(a) Birdchirps')
subplot(2,2,2);plot(s(2,:));
xlabel('time');ylabel('s_{2}');title('(b) Moshi-Moshi')
subplot(2,2,3);plot(x(1,:));
xlabel('time');ylabel('x_{1}');title('(c) Mixed signal ①')
subplot(2,2,4);plot(x(2,:));
xlabel('time');ylabel('x_{2}');title('(d) Mixed signal ②')

```

上記のプログラムを実行すると、元の音声信号と混合された信号の波形が次のように表示される。

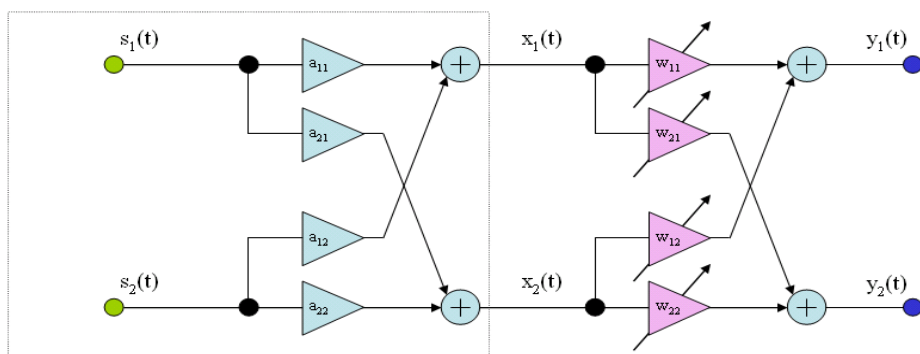


3. モデルの解析

前節では「マイクロフォン信号 $\mathbf{x}(t)$ は既知であるが伝達特性係数 A が未知である、という条件下で、源信号 $\mathbf{s}(t)$ の近似解を求めるには何らかの手法が必要である」と述べた。

本節では、その近似解を求めるために、下図に示すようなニューラルネットワークを利用することを考えてみる。

今ニューラルネットワークの入力を $\mathbf{x}(t) = [x_1(t) \ x_2(t)]^T$ とし、出力を $\mathbf{y}(t) = [y_1(t) \ y_2(t)]^T$ とする。また、ニューラルネットワークの重み行列を $W = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}$ とする。



上図から、ニューラルネットワークの入出力の関係は次式で表現される。

$$\mathbf{y}(t) = W\mathbf{x}(t) \quad (4)$$

式(3)を式(4)に代入すると、式(5)が得られる。

$$\mathbf{y}(t) = W\mathbf{A}\mathbf{s}(t) \quad (5)$$

ここで、式(5-5)において、係数行列 W は自由に調整できることを仮定している。

もし、偶然 W の値が $W = A^{-1}$ と調整された場合、式(5-5)から $\mathbf{y}(t) = \mathbf{s}(t)$ となり、分離信号 $\mathbf{y}(t)$ は信号源 $\mathbf{s}(t)$ と等しくなる。つまり、源信号はマイクロフォン信号から忠実に分離されたことになるが、実際、常にこのような場合になるとは限らない。

今、もう少し現実的に、仮に W の値が $W = PDA^{-1}$ となることを考えよう。ここで、 P は順番置換行列(permutation matrix)と呼ばれ、各行と各列に1に等しい成分が1個ずつあり、その他の成分が0の行列である。例えば、2次元の順番置換行列は $P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ か $P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ のどちらかである。

また、 D は対角行列(diagonal matrix)で、2次元の場合には $D = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix}$ または $D = \begin{bmatrix} 0 & d_1 \\ d_2 & 0 \end{bmatrix}$ と表現される。

上記仮定を考慮して、 $W = PDA^{-1}$ を式(5)に代入すると、分離された信号 $\mathbf{y}(t)$ と信号源 $\mathbf{s}(t)$ との関係は

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} \begin{bmatrix} s_1(t) \\ s_2(t) \end{bmatrix} \quad (6)$$

または

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & d_1 \\ d_2 & 0 \end{bmatrix} \begin{bmatrix} s_1(t) \\ s_2(t) \end{bmatrix} \quad (7)$$

の二通りであることが分かる。

ここで、式(6)に着目すると、 $y_1(t) = d_1 s_1(t)$ また、 $y_2(t) = d_2 s_2(t)$ であり、分離された信号 $\mathbf{y}(t)$ の大きさは対応する対角行列 D の成分によってスケーリングされている可能性がある。また、式(7)から $y_1(t) = d_1 s_2(t)$ 、 $y_2(t) = d_2 s_1(t)$ であり、復元された信号の大きさは D によるスケーリングされるだけでなく、その順番が必ずしも原信号と一致しない。このことを各自実験中に観察してみること。

4. 学習アルゴリズムの設計

前節では、重み行列の候補が $W = PDA^{-1}$ または $W = A^{-1}$ であれば、混合信号が個別に分離ができる、ということを述べた。しかし、実際には、係数 A が未知であり、また重み行列 W を構成する係数の可能な組み合わせは無限にあるため、上述の理想的な解を求めることはほとんど不可能である。本節では、近年、情報理論を駆使して考案された混合信号の分離を行うための自動的な重み W の計算手法について説明する。なお、詳細については、あらかじめ統計情報学を習得している必要があるため、ここでは計算手法の枠組みについて大まかに説明する。

まず、源信号が互いに統計的に独立 ($p_s(\mathbf{s}) = \prod_{i=1}^n p_i(s_i)$ 結合分布の確率密度関数が周辺分布の確率密度関数と等しい) であるとし、独立信号 $\mathbf{s}$ が A によって伝達され、 $\mathbf{x}$ として観測されるとする。基本的に、独立成分解析(ICA: Independent Component Analysis)とは、観測信号 $\mathbf{x}$ のみを用いて、出力信号 $\mathbf{y}$ が限りなく統計的に独立になるような、ニューラルネットワークの再構成することである。すなわち、出力信号 $\mathbf{y}$ の結合分布の確率密度関数がその周辺分布の確率密度関数と等しくなったとき、つまり、

$$p_y(\mathbf{y}) = \prod_{i=1}^n p_i(y_i)$$

を満たすとき、統計的に独立した源信号 $\mathbf{y}$ が推定できた、原信号を復元することになる。

この統計的な独立性を計る情報量としては次式に定義される Kullback-Leibler 情報量が用いられる。

$$D(\mathbf{y} | W) = \int p_y(\mathbf{y}) \log \frac{p_y(\mathbf{y})}{\prod_{i=1}^n p_i(y_i)} d\mathbf{y} \quad (8)$$

ここで、式(5-8)において、 $\mathbf{y}$ の結合分布の確率密度関数とその周辺分布の確率密度関数の関係が $p_y(\mathbf{y}) = \prod_{i=1}^n p_i(y_i)$ となれば、Kullback-Leibler 情報量が $D(\mathbf{y} | W) = \mathbf{0}$ となる。

この情報量が最小になるように重み行列の学習アルゴリズムは $\frac{\partial D(\mathbf{y} | W)}{\partial W} = \mathbf{0}$ で導くことができる。最終的に導かれた逐次型 (on-line) の学習アルゴリズムは

$$\begin{aligned} W(t+1) &= W(t) + \Delta W(t) \\ &= W(t) + \eta(t)[\mathbf{I} - \varphi(\mathbf{y}(t))\mathbf{y}^T(t)]W(t) \end{aligned} \quad (9)$$

となる。上式において、 $\mathbf{I}$ は単位行列であり、 $\eta(t)$ は学習率、また $\varphi(\cdot)$ は Score 関数と呼ばれる。Score 関数 $\varphi(\cdot)$ の成分 $\varphi_i = -\frac{d}{dy_i} \log p_i(y_i) = -\frac{\dot{p}_i(y_i)}{p_i(y_i)}$ は各信号の確率密度関数 $p_i(\cdot)$ に依存することが判っている。

関数 $\varphi(\cdot)$ の近似方法は幾つかが提案されているが、一般的には信号分布の尖（とがり）度を表す kurtosis $\kappa(y) = \frac{E\{y^4\}}{(E\{y^2\})^2} - 3$ を計算し、 $\varphi(\cdot)$ を選択することが多い。

例えば、音声信号 ($\kappa > 0$) の場合には関数

$$\varphi(y) = \tanh(y) \quad (10)$$

が用いられる。また、画像や通信信号 ($\kappa < 0$) の場合には、次の関数を使用する。

$$\varphi(y) = y^3 \quad (11)$$

例題 2：例題1で保存された混合信号を利用して音声分離を行う。但し、学習アルゴリズムは式(5-9)を利用し、 $\varphi(y) = \tanh(y)$ を用いる。確認のため、源信号の尖度

$$\kappa(s) = \frac{E\{s^4\}}{(E\{s^2\})^2} - 3 \text{ を計算してみよう。}$$

プログラム samp2.m <pre> % Separation of the mixed speech signals % filename: samp5_2.m % Jianting Cao, Aug. 10, 2003 % Workspace 変数をすべて削除 clear all </pre>

```

% 混合信号の読み込み
load x;

% 学習回数、学習率、重みの初期値の設定
T=10000;
eta=1.8;
W=eye(2);

% 逐次学習
for t=1:T/home;
    y(:,t)=W*x(:,t);
    W=W+eta/t*[eye(2)-tanh(y(:,t))*y(:,t)']*W;
    W*A
end

% 計算結果の表示
subplot(2,2,1);plot(x(1,:));
xlabel('time');ylabel('x_{1}');title(' (a) Mixed signal ①')
subplot(2,2,2);plot(x(2,:));
xlabel('time');ylabel('x_{2}');title(' (b) Mixed signal ②')
subplot(2,2,3);plot(y(1,:));
xlabel('time');ylabel('y_{1}');title(' (c) Separated signal ①')
subplot(2,2,4);plot(y(2,:));
xlabel('time');ylabel('y_{2}');title(' (d) Separated signal ②')

% 分離した信号を試聴する
clc;input(' 分離した信号① ... ');
sound(y(1,:)/max(y(1,:)));
clc;input(' 分離した信号② ... ');
sound(y(2,:)/max(y(2,:)));

```

5. 画像の復元

前述したように、画像工学・医用画像処理の分野においては、複数の画像が重なったものからある特定の画像を復元することや、ノイズに埋もれた画像から原画雑に復元するという問題がよく現れる。次の例題3では、前節に述べた信号分離の方法を用いて画像の復元を試みる。

例題 3： ホームページ(<http://www.sit.ac.jp/user/cao/Teaching/Teaching.htm> 第6章画像データ)から「testpat1.tif」と「rice.tif」という2つの画像データをダウンロードし、ハードディスクDにあるフォルダMYWORKに保存する。次のプログラムを十分に理解したうえで実行し、画像復元の効果を確かめよう。なお、プログラム中分からないMATLABコマンドがあれば、各自Helpで調べること。

プログラム samp3.m
% Recovered the mixture of images

```

% filename: samp5_3.m
% Jianting Cao, Aug. 10, 2003

% Workspace 変数をすべて削除
clear all

% 元画像データの読み込み
S1=double(imread('testpat1.tif'));
S2=double(imread('rice.tif'));

% 元画像の表示
subplot(3,2,1);image(S1/2);title('(a) Original image ①')
subplot(3,2,2);image(S2/2);title('(b) Original image ②')
hold on

% 2次元原画像データを1次元データに落とす
[I J]=size(S1);
s=[reshape(S1,1,I*J);reshape(S2,1,I*J)];

% 混合係数と混合信号の生成
A=[-0.4689 0.3821; -0.3659 1.0933];
x=A*s;

% 1次元のデータを2次元配列データに戻す
X1=reshape(x(1,:),I,J);
X2=reshape(x(2,:),I,J);

% 混合画像の表示
subplot(3,2,3);image(X1/2);title('(c) Mixed image ①')
subplot(3,2,4);image(X2/2);title('(d) Mixed image ②')
hold on

% 学習データの作成
trainx=x-mean(x')'*ones(1,length(x));
trainx=trainx/max(max(x'));

% 学習回数、学習率、重みの初期値の設定
T=10000;
eta=100;
W=eye(2);

% 逐次学習
for t=1:T;if rem(t,1000)==0;t
    end;home;
    trainy(:,t)=W*trainx(:,t);
    W=W+eta/T*[eye(2)-trainy(:,t).^3*trainy(:,t)']*W;
G=W*A

```

```

end

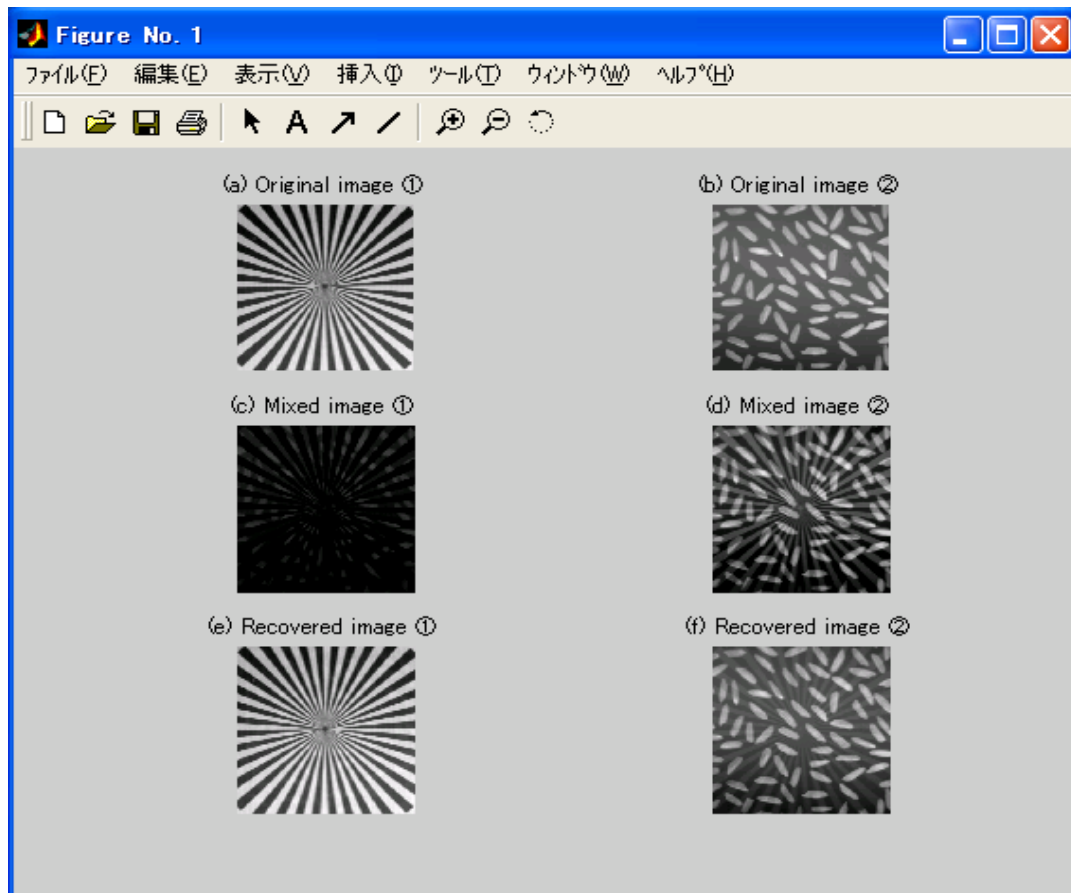
% 学習した復元データの計算
y=W*x;

% 2次元の復元画像データへ変換
Y1=reshape(y(1,:), I, J);
Y2=reshape(y(2,:), I, J);

% 復元画像の表示
scale1=2*G(1,1);scale2=2*G(2,2);
subplot(3,2,5);image(Y1/scale1);title('(e) Recovered image ①')
subplot(3,2,6);image(Y2/scale2);title('(f) Recovered image ②')
hold off

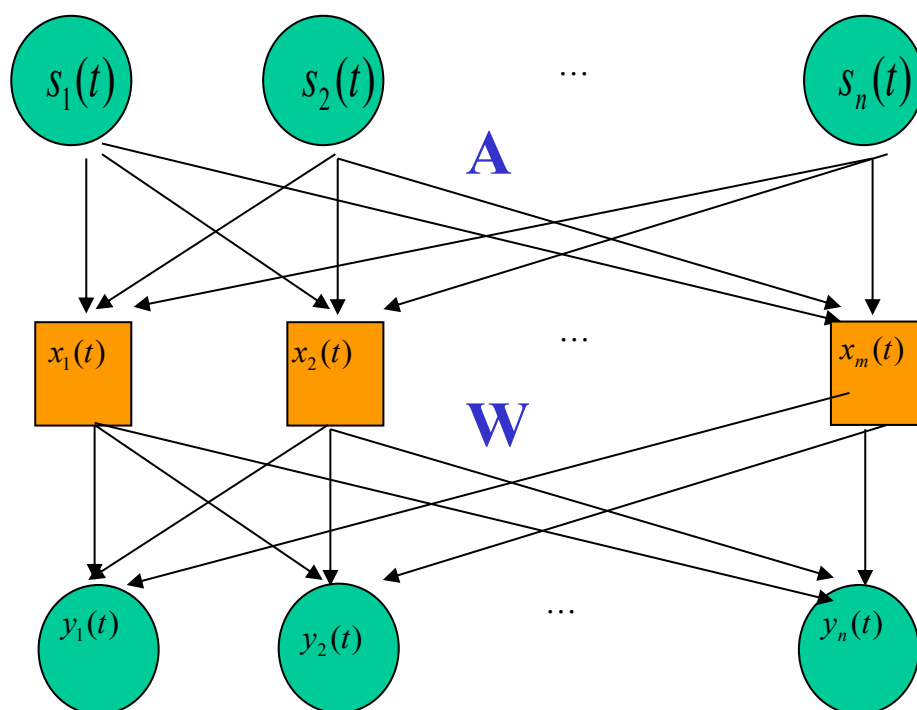
```

上記のプログラムを実行すると、次図のようになる。



3.3 実験課題

1. 一般化された瞬時混合モデルのブロック図は次のように示される。ここで、混合信号の数は信号源の数より多い($m > n$)とする。



このブロック図に基づいて、1) 信号源、2) 混合行列、3) 混合信号、4) 重み行列、5) 分離・復元信号、及びそれらの関係を数式で正しく記述せよ。また、特別な場合、すなわち、 $m=n=2$ で混合行列の値が $A=\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ の場合に、信号分離ができるかどうかについて論じよ。

2. 例題 1 と例題 2 において、一つのノイズ信号源(randn 関数によって生成)を加えた計 3 個の混合信号を分離するようなプログラムを作成せよ。なお、レポートにプログラムと実験結果を示せ。参考のために実行結果の一例を下図に示す。時間的に余裕がある場合はプログラムを function M-ファイルで作成せよ。

