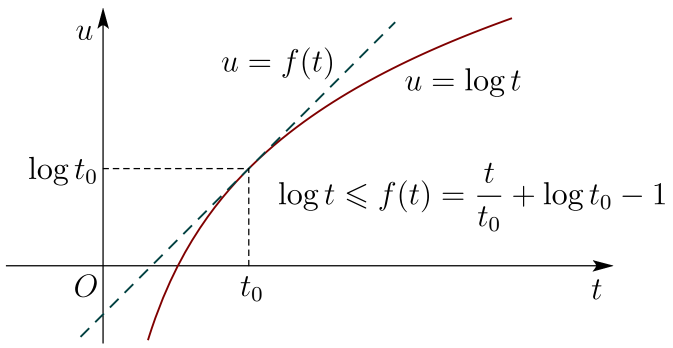
CS 229, Autumn 2016
Problem Set #4 Solutions: Unsupervised learning & RL

Qingxin6174@gmail.com

1. We'll derive the EM updates by maximizing log-likelihood estimation.

$$
\begin{aligned}
\log\left(\prod_{i=1}^{m} p\left(x^{(i)}|\theta\right)\right)p(\theta) &= \log p(\theta) + \sum_{i=1}^{m}\log\sum_{z^{(i)}} p\left(x^{(i)},\, z^{(i)}|\theta\right) \\
&= \log p(\theta) + \sum_{i=1}^{m}\log\sum_{z^{(i)}} Q_i(z^{(i)})\frac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{Q_i(z^{(i)})} \\
&\geqslant \log p(\theta) + \sum_{i=1}^{m}\sum_{z^{(i)}} Q_i(z^{(i)})\log\frac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{Q_i(z^{(i)})}, \qquad (1)
\end{aligned}
$$

"$\geqslant$" is given by the following facts:



Let $t_0 = \sum_{z^{(i)}} Q_i(z^{(i)})\frac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{Q_i(z^{(i)})}$, and then

$$
\begin{aligned}
&\sum_{z^{(i)}} Q_i(z^{(i)})\log\frac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{Q_i(z^{(i)})} \\
\leqslant &\sum_{z^{(i)}} Q_i(z^{(i)})\, f\left(\frac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{Q_i(z^{(i)})}\right) \\
= &\log t_0.
\end{aligned}
$$

"$=$" holds true if and only if $\forall z^{(i)}$, $\dfrac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{Q_i(z^{(i)})} = t_0$, note that t_0 is constant, which means $Q_i(z^{(i)}) \propto p\left(x^{(i)},\, z^{(i)}|\theta\right)$, since $\sum_{z^{(i)}} Q_i(z^{(i)}) = 1$, this implies that

$$
Q_i(z^{(i)}) = \frac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{\sum_{z^{(i)}} p\left(x^{(i)},\, z^{(i)}|\theta\right)} = p\left(z^{(i)}|x^{(i)},\, \theta\right), \qquad (2)
$$

which is the E-step.
For the M-step, we update θ by maximizing the lower bound

$$
\theta = \arg\max_{\theta}\left(\log p(\theta) + \sum_{i=1}^{m}\sum_{z^{(i)}} Q_i(z^{(i)})\log\frac{p\left(x^{(i)},\, z^{(i)}|\theta\right)}{Q_i(z^{(i)})}\right). \qquad (3)
$$

Define fomula (1) as $J(Q,\, \theta)$, and suppose $\theta^{(n-1)}$ and $\theta^{(n)}$ are the parameters from two successive iterations of EM. We will prove that $J(Q^{(n)},\, \theta^{(n)}) \geqslant J(Q^{(n-1)},\, \theta^{(n-1)})$, which shows EM always monotonically improves the log-likelihood.
Fomula (2) $Q^{(n)} \longleftarrow \theta^{(n-1)}$ is choosing $Q^{(n)}$ to make $J(Q^{(n)},\, \theta) = \max_Q J(Q,\, \theta)$, and fomula (3) $\theta^{(n)} = \arg\max_\theta J(Q^{(n)},\, \theta)$ is choosing $\theta^{(n)}$ to make $J(Q^{(n)},\, \theta^{(n)}) = \max_\theta J(Q^{(n)},\, \theta)$, thus we have the following derivation:

$$
J(Q^{(n)},\, \theta^{(n)}) = \max_{\theta} J(Q^{(n)},\, \theta) = \max_{Q,\,\theta} J(Q,\, \theta) \geqslant \max_{\theta} J(Q^{(n-1)},\, \theta) = J(Q^{(n-1)},\, \theta^{(n-1)}),
$$

this implies that the nth iteration $\left(\prod_{i=1}^{m} p\left(x^{(i)}|\theta^{(n)}\right)\right)p(\theta^{(n)}) = e^{J(Q^{(n)},\theta^{(n)})}$ increase monotonically, the EM update rules will get the maximum likelihood estimation, and this algorithm will converge.

2. (a) i. $x^{(pr)}$ can be written as $x^{(pr)} = y^{(pr)} + z^{(pr)} + \varepsilon^{(pr)}$, where $y^{(pr)} \sim \mathcal{N}\left(\mu_p,\, \sigma_p^2\right)$, $z^{(pr)} \sim \mathcal{N}\left(\nu_r,\, \tau_r^2\right)$ and $\varepsilon^{(pr)} \sim \mathcal{N}\left(0,\, \sigma^2\right)$ are independent, then we have:

$$\begin{aligned}
E\left[x^{(pr)}\right] &= E\left[y^{(pr)}\right] + E\left[z^{(pr)}\right] + E\left[\varepsilon^{(pr)}\right] = \mu_p + \nu_r, \\
\operatorname{Var}\left(x^{(pr)}\right) &= \operatorname{Var}\left(y^{(pr)}\right) + \operatorname{Var}\left(z^{(pr)}\right) + \operatorname{Var}\left(\varepsilon^{(pr)}\right) = \sigma_p^2 + \tau_r^2 + \sigma^2, \\
\operatorname{Cov}\left(y^{(pr)},\, x^{(pr)}\right) &= \operatorname{Cov}\left(y^{(pr)},\, y^{(pr)}\right) + \operatorname{Cov}\left(y^{(pr)},\, z^{(pr)}\right) + \operatorname{Cov}\left(y^{(pr)},\, \varepsilon^{(pr)}\right) = \sigma_p^2, \\
\operatorname{Cov}\left(z^{(pr)},\, x^{(pr)}\right) &= \operatorname{Cov}\left(z^{(pr)},\, y^{(pr)}\right) + \operatorname{Cov}\left(z^{(pr)},\, z^{(pr)}\right) + \operatorname{Cov}\left(z^{(pr)},\, \varepsilon^{(pr)}\right) = \tau_r^2,
\end{aligned}$$

then we get $p\left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right)$'s multivariate Gaussian density:

$$y^{(pr)},\, z^{(pr)},\, x^{(pr)} \sim \mathcal{N}\left(\mu,\, \Sigma\right) = \mathcal{N}\left(\begin{bmatrix} \mu_p \\ \nu_r \\ \mu_p + \nu_r \end{bmatrix},\, \begin{bmatrix} \sigma_p^2 & 0 & \sigma_p^2 \\ 0 & \tau_r^2 & \tau_r^2 \\ \sigma_p^2 & \tau_r^2 & \sigma_p^2 + \tau_r^2 + \sigma^2 \end{bmatrix}\right).$$

ii. As $y^{(pr)}$'s and $z^{(pr)}$'s are all latent random variables, the log-likelihood function is

$$\begin{aligned}
\log \prod_{p,\, r} p\left(x^{(pr)};\, \mu_p,\, \sigma_p^2,\, \nu_r,\, \tau_r^2\right) &= \sum_{p,\, r} \log \sum_{y^{(pr)},\, z^{(pr)}} Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right) \frac{p\left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right)}{Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right)} \\
&\geqslant \sum_{p,\, r} \sum_{y^{(pr)},\, z^{(pr)}} Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right) \log \frac{p\left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right)}{Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right)},
\end{aligned}$$

which implies $Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right) = \dfrac{p\left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right)}{\sum_{y^{(pr)},\, z^{(pr)}} p\left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right)} = p\left(y^{(pr)},\, z^{(pr)} | x^{(pr)}\right)$.

Let $\mathcal{N}\left(\begin{bmatrix} \mu_{yz} \\ \mu_x \end{bmatrix},\, \begin{bmatrix} \Sigma_{yz,yz} & \Sigma_{yz,x} \\ \Sigma_{x,yz} & \Sigma_{x,x} \end{bmatrix}\right) := \mathcal{N}\left(\left[\begin{array}{c} \mu_p \\ \nu_r \\ \hline \mu_p + \nu_r \end{array}\right],\, \left[\begin{array}{cc|c} \sigma_p^2 & 0 & \sigma_p^2 \\ 0 & \tau_r^2 & \tau_r^2 \\ \hline \sigma_p^2 & \tau_r^2 & \sigma_p^2 + \tau_r^2 + \sigma^2 \end{array}\right]\right)$,

according to the Gaussian Factor Analysis, we obtain the E-step:

$$\begin{aligned}
&Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right) = p\left(y^{(pr)},\, z^{(pr)} | x^{(pr)}\right) \\
&= \mathcal{N}\left(\mu_{yz} + \Sigma_{yz,x}\Sigma_{x,x}^{-1}\left(x^{(pr)} - \mu_p - \nu_r\right),\, \Sigma_{yz,yz} - \Sigma_{yz,x}\Sigma_{x,x}^{-1}\Sigma_{x,yz}\right) \\
&= \mathcal{N}\left(\begin{bmatrix} \mu_p \\ \nu_r \end{bmatrix} + \frac{x^{(pr)} - \mu_p - \nu_r}{\sigma_p^2 + \tau_r^2 + \sigma^2}\begin{bmatrix} \sigma_p^2 \\ \tau_r^2 \end{bmatrix},\, \begin{bmatrix} \sigma_p^2 & 0 \\ 0 & \tau_r^2 \end{bmatrix} - \frac{1}{\sigma_p^2 + \tau_r^2 + \sigma^2}\begin{bmatrix} \sigma_p^2 \\ \tau_r^2 \end{bmatrix}\begin{bmatrix} \sigma_p^2 & \tau_r^2 \end{bmatrix}\right).
\end{aligned}$$

if we define

$$\mathcal{N}\left(\begin{bmatrix} \mu_y \\ \mu_z \end{bmatrix},\, \begin{bmatrix} \Sigma_{y,y} & \Sigma_{y,z} \\ \Sigma_{z,y} & \Sigma_{z,z} \end{bmatrix}\right) := Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right),$$

we obtain the parameters of E-step:

$$\begin{bmatrix} \mu_y \\ \mu_z \end{bmatrix} \longleftarrow \begin{bmatrix} \dfrac{\left(x^{(pr)} - \nu_r - \mu_p\right)\sigma_p^2}{\sigma^2 + \sigma_p^2 + \tau_r^2} + \mu_p \\ \dfrac{\left(x^{(pr)} - \nu_r - \mu_p\right)\tau_r^2}{\sigma^2 + \sigma_p^2 + \tau_r^2} + \nu_r \end{bmatrix}, \tag{4}$$

$$\begin{bmatrix} \Sigma_{y,y} & \Sigma_{y,z} \\ \Sigma_{z,y} & \Sigma_{z,z} \end{bmatrix} \longleftarrow \begin{bmatrix} \sigma_p^2 - \dfrac{\sigma_p^4}{\sigma^2 + \sigma_p^2 + \tau_r^2} & -\dfrac{\sigma_p^2\tau_r^2}{\sigma^2 + \sigma_p^2 + \tau_r^2} \\ -\dfrac{\sigma_p^2\tau_r^2}{\sigma^2 + \sigma_p^2 + \tau_r^2} & \tau_r^2 - \dfrac{\tau_r^4}{\sigma^2 + \sigma_p^2 + \tau_r^2} \end{bmatrix}. \tag{5}$$

(b) Let $E_Q\xi$ denote the expectations of random variable ξ with respect to density $Q_{pr}\left(y^{(pr)},\, z^{(pr)}\right)$. For the M-step, we update $\mu_p,\, \sigma_p^2,\, \nu_r,\, \tau_r^2$ by maximizing the lower bound with Q_{pr} be fixed, the superscript "fixed" means their parameters are not variables. Let $X := \left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right)^T$.

$$
\begin{aligned}
\mu_p,\, \sigma_p^2,\, \nu_r,\, \tau_r^2 &= \arg \max_{\mu_p, \sigma_p^2, \nu_r, \tau_r^2} \sum_{p,\, r} \sum_{y^{(pr)},\, z^{(pr)}} Q_{pr}^{\text{fixed}}\left(y^{(pr)},\, z^{(pr)}\right) \log \frac{p\left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right)}{Q_{pr}^{\text{fixed}}\left(y^{(pr)},\, z^{(pr)}\right)} \\
&= \arg \max_{\mu_p, \sigma_p^2, \nu_r, \tau_r^2} \sum_{p,\, r} E_{Q^{\text{fixed}}} \log p\left(y^{(pr)},\, z^{(pr)},\, x^{(pr)}\right) \\
&= \arg \max_{\mu_p, \sigma_p^2, \nu_r, \tau_r^2} \sum_{p,\, r} E_{Q^{\text{fixed}}} \left[-\frac{3}{2}\log 2\pi - \frac{1}{2}\log|\Sigma| - \frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)\right] \\
&= \arg \max_{\mu_p, \sigma_p^2, \nu_r, \tau_r^2} \sum_{p,\, r} E_{Q^{\text{fixed}}} \left[-\frac{1}{2}\log \sigma_p^2\tau_r^2 - \frac{\left(y^{(pr)}-\mu_p\right)^2}{2\sigma_p^2} - \frac{\left(z^{(pr)}-\nu_r\right)^2}{2\tau_r^2}\right] \\
&= \arg \min_{\mu_p, \sigma_p^2, \nu_r, \tau_r^2} \sum_{p,\, r} \log \sigma_p^2\tau_r^2 + E_{Q^{\text{fixed}}} \left[\frac{\left(y^{(pr)}-\mu_p\right)^2}{\sigma_p^2} + \frac{\left(z^{(pr)}-\nu_r\right)^2}{\tau_r^2}\right] \\
&= \arg \min_{\mu_p, \sigma_p^2, \nu_r, \tau_r^2} \sum_{p,\, r} \log \sigma_p^2\tau_r^2 + \frac{1}{\sigma_p^2}\left(E_{Q^{\text{fixed}}}\left[\left(y^{(pr)}\right)^2\right] - 2\mu_p E_{Q^{\text{fixed}}}\left[y^{(pr)}\right] + \mu_p^2\right) \\
&\qquad\qquad + \frac{1}{\tau_r^2}\left(E_{Q^{\text{fixed}}}\left[\left(z^{(pr)}\right)^2\right] - 2\nu_r E_{Q^{\text{fixed}}}\left[z^{(pr)}\right] + \nu_r^2\right) \\
&= \arg \min_{\mu_p, \sigma_p^2, \nu_r, \tau_r^2} \sum_{p,\, r} \log \sigma_p^2\tau_r^2 + \frac{1}{\sigma_p^2}\left(\Sigma_{y,y}^{\text{fixed}} + \left(\mu_y^{\text{fixed}}\right)^2 - 2\mu_p\mu_y^{\text{fixed}} + \mu_p^2\right) \\
&\qquad\qquad + \frac{1}{\tau_r^2}\left(\Sigma_{z,z}^{\text{fixed}} + \left(\mu_z^{\text{fixed}}\right)^2 - 2\nu_r\mu_z^{\text{fixed}} + \nu_r^2\right),
\end{aligned}
$$

if we define

$$
\begin{aligned}
J\left(\mu_p,\, \sigma_p^2,\, \nu_r,\, \tau_r^2\right) = \sum_{p,\, r} \log \sigma_p^2\tau_r^2 &+ \frac{1}{\sigma_p^2}\left(\Sigma_{y,y}^{\text{fixed}} + \left(\mu_y^{\text{fixed}}\right)^2 - 2\mu_p\mu_y^{\text{fixed}} + \mu_p^2\right) \\
&+ \frac{1}{\tau_r^2}\left(\Sigma_{z,z}^{\text{fixed}} + \left(\mu_z^{\text{fixed}}\right)^2 - 2\nu_r\mu_z^{\text{fixed}} + \nu_r^2\right),
\end{aligned}
$$

setting derivatives with respect to parameters $\mu_p,\, \sigma_p^2,\, \nu_r,\, \tau_r^2$ to 0, we have

$$
0 \overset{\text{let}}{=} \frac{\partial J}{\partial \mu_p} = \sum_r \frac{2}{\sigma_p^2}\left(\mu_p - \mu_y^{\text{fixed}}\right) \;\Rightarrow\; \mu_p \longleftarrow \frac{1}{R}\sum_{r=1}^{R} \mu_y, \tag{6}
$$

$$
\begin{aligned}
0 \overset{\text{let}}{=} \frac{\partial J}{\partial \sigma_p^2} &= \sum_r \frac{1}{\sigma_p^2} - \frac{1}{\sigma_p^4}\left(\Sigma_{y,y}^{\text{fixed}} + \left(\mu_y^{\text{fixed}}\right)^2 - 2\mu_p\mu_y^{\text{fixed}} + \mu_p^2\right) \\
&\Rightarrow\; \sigma_p^2 \longleftarrow \frac{1}{R}\sum_{r=1}^{R} \Sigma_{y,y} + \mu_y^2 - 2\mu_p\mu_y + \mu_p^2,
\end{aligned} \tag{7}
$$

$$
0 \overset{\text{let}}{=} \frac{\partial J}{\partial \nu_r} = \sum_p \frac{2}{\tau_r^2}\left(\nu_r - \mu_z^{\text{fixed}}\right) \;\Rightarrow\; \nu_r \longleftarrow \frac{1}{P}\sum_{p=1}^{P} \mu_z, \tag{8}
$$

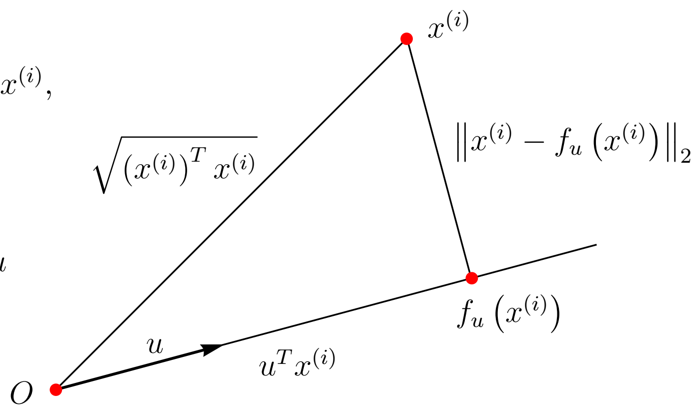
$$
\begin{aligned}
0 \overset{\text{let}}{=} \frac{\partial J}{\partial \tau_r^2} &= \sum_p \frac{1}{\tau_r^2} - \frac{1}{\tau_r^4}\left(\Sigma_{z,z}^{\text{fixed}} + \left(\mu_z^{\text{fixed}}\right)^2 - 2\nu_r\mu_z^{\text{fixed}} + \nu_r^2\right) \\
&\Rightarrow\; \tau_r^2 \longleftarrow \frac{1}{P}\sum_{p=1}^{P} \Sigma_{z,z} + \mu_z^2 - 2\nu_r\mu_z + \nu_r^2.
\end{aligned} \tag{9}
$$

The EM algorithm repeat in order from equation (4) to (9) until convengence.

3. As shown in the cartoon, we have

$$\left\|x^{(i)} - f_u\left(x^{(i)}\right)\right\|_2^2 + u^T x^{(i)} \left(x^{(i)}\right)^T u = \left(x^{(i)}\right)^T x^{(i)},$$

$$\begin{aligned}
&\arg\min_{u:u^Tu=1} \sum_{i=1}^m \left\|x^{(i)} - f_u\left(x^{(i)}\right)\right\|_2^2 \\
&= \arg\min_{u:u^Tu=1} \sum_{i=1}^m \left(x^{(i)}\right)^T x^{(i)} - u^T x^{(i)} \left(x^{(i)}\right)^T u \\
&= \arg\max_{u:u^Tu=1} u^T \left(\sum_{i=1}^m x^{(i)} \left(x^{(i)}\right)^T\right) u.
\end{aligned}$$



4. Here is the MATLAB code filling in `bellsej.m`:

```
for iter=1:length(anneal)
    m = size(mix, 1);
    order = randperm(m);
    for i = 1:m
        x = mix(order(i), :)';
        W = W + anneal(iter) * ((1 - 2./(1+exp(-W*x)))*x' + inv(W'));
    end
end
S = mix * W';
```

and matrix $W = \begin{pmatrix} 31.1334 & 10.0479 & 11.7648 & -6.0910 & -12.0518 \\ 8.4541 & 15.5896 & -1.4624 & -12.5795 & 5.0100 \\ 5.0013 & -4.4380 & 11.5711 & 9.0372 & -8.5817 \\ -5.7731 & -0.3350 & -2.7253 & 4.9096 & 1.0267 \\ -0.3840 & 11.3611 & 5.6962 & 5.6670 & 19.0549 \end{pmatrix}$.

The visualization of signals are shown in Figure 1.

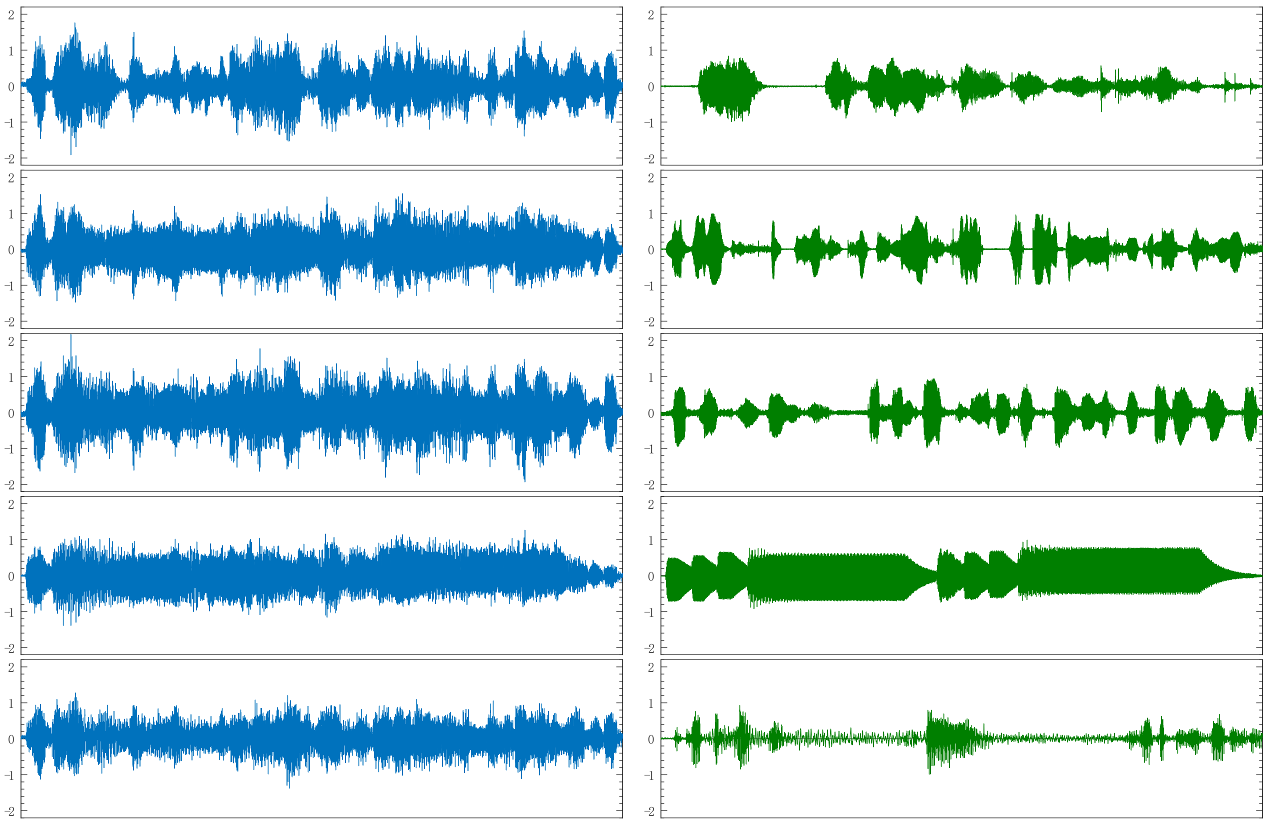


Fig. 1: Mixed sources vs. unmixed signals.

5. (a) Let's assume that $a_1^* = \arg\max_{a\in A} \sum_{s'\in S} P_{sa}(s') V_1(s')$, and $a_2^* = \arg\max_{a\in A} \sum_{s'\in S} P_{sa}(s') V_2(s')$. Noticing that the conclusion is a rotating symmetric formula, we can assume that $V_1'(s) \geqslant V_2'(s)$, then we have

$$\begin{aligned}
\|B(V_1) - B(V_2)\|_\infty &= \max_{s\in S} |V_1'(s) - V_2'(s)| \\
&= \max_{s\in S} \left| \gamma \max_{a\in A} \sum_{s'\in S} P_{sa}(s') V_1(s') - \gamma \max_{a\in A} \sum_{s'\in S} P_{sa}(s') V_2(s') \right| \\
&= \gamma \max_{s\in S} \left| \sum_{s'\in S} P_{sa_1^*}(s') V_1(s') - \sum_{s'\in S} P_{sa_2^*}(s') V_2(s') \right| \\
&\leqslant \gamma \max_{s\in S} \left| \sum_{s'\in S} P_{sa_1^*}(s') V_1(s') - \sum_{s'\in S} P_{sa_1^*}(s') V_2(s') \right| \\
&\leqslant \gamma \max_{s\in S} \max_{a\in A} \left| \sum_{s'\in S} P_{sa}(s') V_1(s') - \sum_{s'\in S} P_{sa}(s') V_2(s') \right| \\
&\leqslant \gamma \max_{s\in S} \max_{a\in A} \sum_{s'\in S} P_{sa}(s') |V_1(s') - V_2(s')| \\
&\leqslant \gamma \max_{s\in S} \max_{a\in A} \max_{s'\in S} |V_1(s') - V_2(s')| \\
&= \gamma \max_{s'\in S} |V_1(s') - V_2(s')| \\
&= \gamma \|V_1 - V_2\|_\infty .
\end{aligned}$$

(b) Let's assume that B has at least one fixed point, for example: $B(V_1) = V_1$ and $B(V_2) = V_2$, where $V_1 \neq V_2$, then

$$\|V_1 - V_2\|_\infty = \|B(V_1) - B(V_2)\|_\infty \leqslant \gamma \|V_1 - V_2\|_\infty < \|V_1 - V_2\|_\infty ,$$

which is contradictory. Then B has at most one fixed point.

6. (a) Here is the MATLAB code filling in `control.m`:

```
...Code has already been given...
reward = zeros(NUM_STATES, 1); % Initializations
value = 0.1 * rand(NUM_STATES, 1);
reward_counts = zeros(NUM_STATES, 2);
transition_probabilities = ones(NUM_STATES, NUM_STATES, 2) / NUM_STATES;
transition_counts = zeros(NUM_STATES, NUM_STATES, 2);
number_of_consecutive_no_learning_trials = 0;
while number_of_consecutive_no_learning_trials < NO_LEARNING_THRESHOLD
    % Choose action 1 or 2
    score_1 = transition_probabilities(state,:,1) * value;
    score_2 = transition_probabilities(state,:,2) * value;
    action = 2 - (score_1>score_2) - ((score_1==score_2) & (rand<0.5));
    % Get the next state by simulating the dynamics
    [x, x_dot, theta, theta_dot] = ...
        cart_pole(action, x, x_dot, theta, theta_dot);
    time = time + 1; % Increment simulation time
    % Get the state number corresponding to new state vector
    new_state = get_state(x, x_dot, theta, theta_dot);
    R = -1*(new_state == NUM_STATES); % Reward function
    % Perform updates
```

```
    % Perform updates
    transition_counts(state, new_state, action) = ...
        transition_counts(state, new_state, action) + 1;
    reward_counts(new_state,1) = reward_counts(new_state,1) + R;
    reward_counts(new_state,2) = reward_counts(new_state,2) + 1;
    % Recompute MDP model whenever pole falls
    % Compute the value function V for the new model
    if new_state == NUM_STATES
        % Update MDP model
        for action_idx = 1:2 % Update transition_probabilities
            total = sum(transition_counts(:,:,action_idx),2);
            idx = total > 0;
            transition_probabilities(idx,:,action_idx) = ...
                transition_counts(idx,:,action_idx) ./ ...
                (total(idx,:) * ones(1,NUM_STATES));
        end
        idx = reward_counts(:,2) > 0; % Update reward
        reward(idx) = reward_counts(idx,1) ./ reward_counts(idx,2);
        % Perform value iteration using the new estimated model for the MDP
        iter = 0; value_diff = 1;
        while value_diff >= TOLERANCE
            iter = iter + 1;
            new_value = reward + GAMMA * ...
                max([transition_probabilities(:,:,1) * value, ...
                     transition_probabilities(:,:,2) * value], [], 2);
            value_diff = max(abs(value - new_value));
            value = new_value;
        end
        number_of_consecutive_no_learning_trials = ...
            (iter == 1) * (number_of_consecutive_no_learning_trials + 1);
    end
    ...Code has already been given...
end
```

239 trials were needed before the algorithm converged.

(b) Figure 2 shows the learning curve.

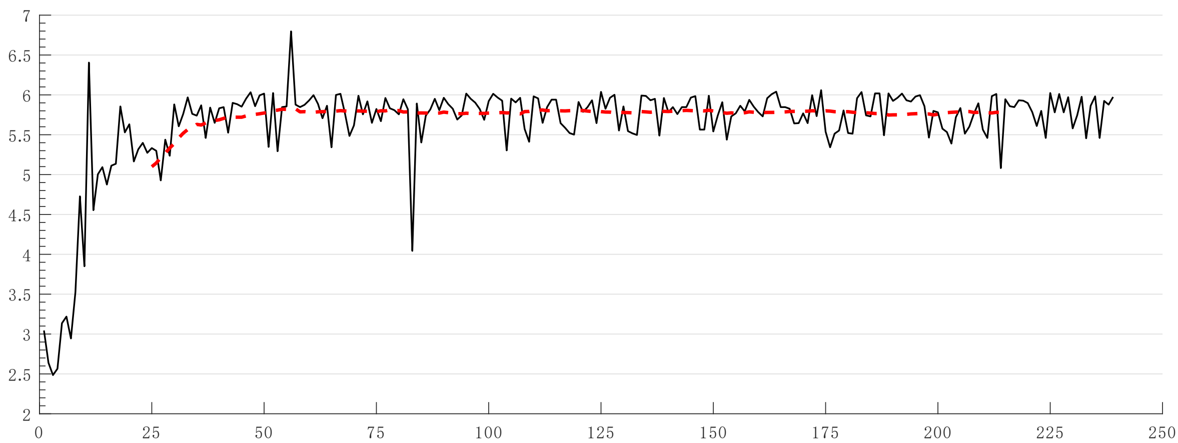


Fig. 2: Learning curve.