DISC Course: Multi-agent Network Dynamics and Games

Assignment 2

Hai Zhu

Delft University of Technology

h.zhu@tudelft.nl

March 18, 2018

E2.02 Determine the convergence rate of the Banach iteration.

Solution: Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be a ℓ-contractive mapping ($\ell \in [0, 1)$). The Banach iteration approximates a fixed point of T by the following iteration

$$x(k+1) = T(x(k)) \tag{1}$$

Then according to Banach's fixed point theorem, T admits a unique fixed-point $x^\star$, i.e. $x^\star = T(x^\star)$.

We have proved the Banach's theorem in last assignment. Now we determine the convergence rate of the Banach iteration.

Since T is ℓ-contraction, we have

$$d(T(x), T(y)) \leq \ell d(x, y) \tag{2}$$

Thus

$$\begin{aligned} d(x_{n+1}, x^\star) &= d(T(x_n), T(x^\star)) \\ &\leq \ell d(x_n, x^\star) \end{aligned} \tag{3}$$

and

$$\begin{aligned} d(x_n, x^\star) &\leq \ell d(x_{n-1}, x^\star) \\ &\leq \ell^2 d(x_{n-2}, x^\star) \\ &\leq \dots \\ &\leq \ell^n d(x_0, x^\star), \quad n = 1, 2, \dots \end{aligned} \tag{4}$$

which determines the rate of convergence of the Banach iteration. Note that $\ell \in [0, 1)$ in the above equations. Hence, the convergence rate of the Banach iteration is linear.

E2.06 Let $A : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ be a maximally monotone mapping. Show that $\text{zer}(A+B) = \text{fix}(\mathcal{J}_{\gamma A} \circ (\text{Id} - \gamma B)), \forall \gamma > 0$.

Proof. Suppose $x^\star$ is the fixed point, i.e., $x^\star = \mathcal{J}_{\gamma A} \circ (\text{Id} - \gamma B)(x^\star)$. Thus

$$\begin{aligned} x^\star = \mathcal{J}_{\gamma A} \circ (\text{Id} - \gamma B)(x^\star) &\iff x^\star = (\text{Id} + \gamma A)^{-1} \circ (\text{Id} - \gamma B)(x^\star) \\ &\iff (\text{Id} + \gamma A)(x^\star) = (\text{Id} - \gamma B)(x^\star) \\ &\iff \gamma(A(x^\star) + B(x^\star)) = 0 \\ &\iff A(x^\star) + B(x^\star) = 0 \\ &\iff x^\star \in \text{zer}(A + B) \end{aligned} \tag{5}$$

Hence, $\text{zer}(A + B) = \text{fix}(\mathcal{J}_{\gamma A} \circ (\text{Id} - \gamma B)), \forall \gamma > 0$.

□

E2.09

Solution: Since $J_1(x) = x_1 x_2$, $J_2(x) = -x_1 x_2$ and $\mathcal{X}_1(\cdot) = \mathcal{X}_1(\cdot) = \mathbb{R}$, we have

$$\text{proj}_{\mathcal{X}_1} = \text{proj}_{\mathcal{X}_2} = \text{Id} \tag{6}$$

and

$$F(x) = \begin{bmatrix} \nabla_{x_1} J_1(x_1, x_2) \\ \nabla_{x_2} J_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_2 \\ -x_1 \end{bmatrix} \tag{7}$$

According to the forward-backward algorithm

$$x(k+1) = \text{proj}_\Omega(x(k) - \epsilon F(x(k))) \tag{8}$$

we have

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} x_1(k) - \epsilon(x_2(k)) \\ x_2(k) + \epsilon x_1(k) \end{bmatrix} = \begin{bmatrix} 1 & -\epsilon \\ \epsilon & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \tag{9}$$

Denote

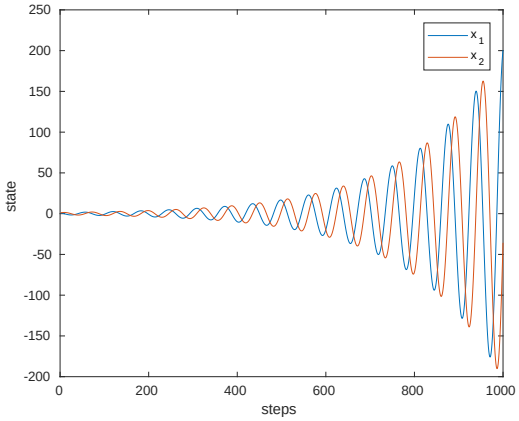
$$T = \begin{bmatrix} 1 & -\epsilon \\ \epsilon & 1 \end{bmatrix} \tag{10}$$

and thus

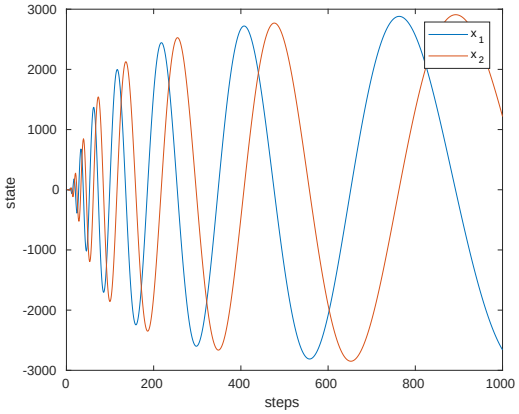
$$x(k+1) = T(x(k)) = Tx(k) \tag{11}$$

Since $\Lambda(T) = 1 \pm \epsilon \text{i}$. The above mapping is not contractive and thus cannot converge to a fixed point.

To simulate the iteration, we consider two cases: with constant step size and vanishing step size in the following part. In both two cases, the starting point is chosen to be $x_1(0) = x_2(0) = 1$. The maximum number of iteration is set as 1×10^3. Case (a) is with constant step size $\epsilon = 0.1$ and case (b) is with vanishing step size $\epsilon = \min\{1, \frac{10}{k}\}$. The following two figures show the Simulation results. It can be seen that in both cases, the solution does not converge.



(a) constant step size $\epsilon = 0.1$



(b) vanishing step size $\epsilon = \min\{1, \frac{10}{k}\}$

Matlab Code for Simulation

```
%% Exercise E2.09
% Simulation of forward-backward algorithm
clear all;
clc;

% Initial configurations
x0 = [1, 1];            % initial condition
xk = x0;                % state at the moment
xN = x0;                % iteration history
maxIter = 1000;         % maximum iteration

%% Simulation with constant step size
% iteration
eps = 0.1;
T = [1, -eps; eps, 1];
k = 1;
while (k < maxIter)
    xk = (T * xk')';    % iteration step
    xN = [xN; xk];
    k = k+1;
end
N = size(xN(:,1));      % total iteration steps
% plotting state history
figure;
hold on;
box on;
xlabel('steps');
ylabel('state');
plot(1:N, xN(:,1), 1:N, xN(:,2));
legend('x_1','x_2');
saveas(gca,'E209_cons_state','pdf');
system('pdfcrop E209_cons_state.pdf E209_cons_state.pdf');
```

```
% plotting trajectory
figure;
hold on;
box on;
xlabel('x_1');
ylabel('x_2');
plot(xN(:,1), xN(:,2));
saveas(gca,'E209_cons_tra','pdf');
system('pdfcrop E209_cons_tra.pdf E209_cons_tra.pdf');
% plotting cost function history
J1N = xN(:,1).*xN(:,2);
J2N = -xN(:,1).*xN(:,2);
figure;
hold on;
box on;
xlabel('steps');
ylabel('cost function');
plot(1:N, J1N, 1:N, J2N);
legend('J_1','J_2');
saveas(gca,'E209_cons_cost','pdf');
system('pdfcrop E209_cons_cost.pdf E209_cons_cost.pdf');

%% Simulation with vanishing step size
xk = x0; xN = x0;
k = 1;
while (k < maxIter)
    eps = min(1, 10 / k);
    T = [1, -eps; eps, 1];
    xk = (T * xk')';    % iteration step
    xN = [xN; xk];
    k = k+1;
end
N = size(xN(:,1));      % total iteration steps
% plotting state history
figure;
hold on;
box on;
xlabel('steps');
ylabel('state');
plot(1:N, xN(:,1), 1:N, xN(:,2));
legend('x_1','x_2');
saveas(gca,'E209_van_state','pdf');
system('pdfcrop E209_van_state.pdf E209_van_state.pdf');
% plotting trajectory
figure;
hold on;
```

```
box on;
xlabel('x_1');
ylabel('x_2');
plot(xN(:,1), xN(:,2));
saveas(gca,'E209_van_tra','pdf');
system('pdfcrop E209_van_tra.pdf E209_van_tra.pdf');
% plotting cost function history
J1N = xN(:,1).*xN(:,2);
J2N = -xN(:,1).*xN(:,2);
figure;
hold on;
box on;
xlabel('steps');
ylabel('cost function');
plot(1:N, J1N, 1:N, J2N);
legend('J_1','J_2');
saveas(gca,'E209_van_cost','pdf');
system('pdfcrop E209_van_cost.pdf E209_van_cost.pdf');
```

E2.10

Solution: Since $J_1(x) = \frac{1}{2}x_1^2 + x_1x_2$, $J_2(x) = -x_1x_2$ and $\mathcal{X}_1(\cdot) = \mathcal{X}_1(\cdot) = \mathbb{R}$, we have

$$\text{proj}_{\mathcal{X}_1} = \text{proj}_{\mathcal{X}_2} = \text{Id} \tag{12}$$

and

$$F(x) = \begin{bmatrix} \nabla_{x_1} J_1(x_1, x_2) \\ \nabla_{x_2} J_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ -x_1 \end{bmatrix} \tag{13}$$

According to the forward-backward algorithm

$$x(k+1) = \text{proj}_{\Omega}(x(k) - \epsilon F(x(k))) \tag{14}$$

we have

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} x_1(k) - \epsilon(x_1(k) + x_2(k)) \\ x_2(k) + \epsilon x_1(k) \end{bmatrix} = \begin{bmatrix} 1-\epsilon & -\epsilon \\ \epsilon & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \tag{15}$$

Denote

$$T = \begin{bmatrix} 1-\epsilon & -\epsilon \\ \epsilon & 1 \end{bmatrix} \tag{16}$$

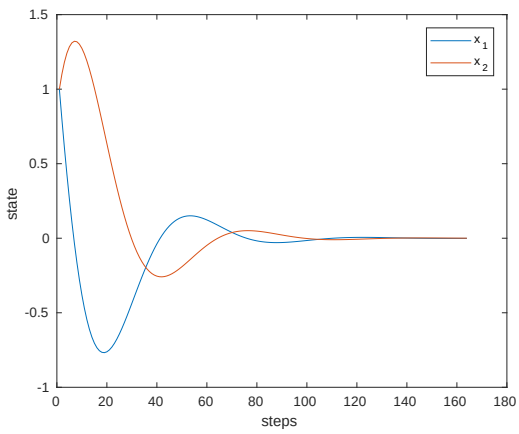
and thus

$$x(k+1) = T(x(k)) = Tx(k) \tag{17}$$

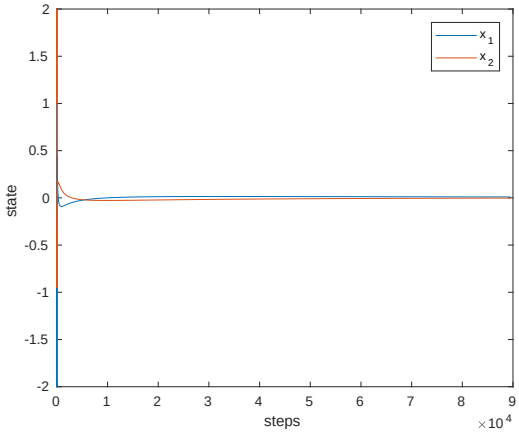
Since $\Lambda(T) = 1 - \frac{\epsilon}{2} \pm \frac{\sqrt{3}}{2}\epsilon \text{i}$, if we choose $\epsilon \in (0, 1)$, then the above iteration becomes the Banach iteration. Hence, it will converge to the fixed point from any starting point.

To simulate the iteration, we consider two cases: with constant step size and vanishing step size in the following part. In both two cases, the starting point is chosen to be $x_1(0) =$

$x_2(0) = 1$. The tolerance error of iteration is set as $\varepsilon = 1 \times 10^{-2}$. Case (a) is with constant step size $\epsilon = 0.1$ and case (b) is with vanishing step size $\epsilon = \frac{1}{k}$. The following two figures show the Simulation results. It can be seen that in both cases, the solution converges. Besides, the convergence of the iteration with constant size is faster than that with a vanishing step size.



(a) constant step size $\epsilon = 0.1$



(b) vanishing step size $\epsilon = \frac{1}{k}$

Matlab Code for Simulation

```
%% Exercise E2.10
% Simulation of forward-backward algorithm
clear all;
clc;

% Initial configurations
x0 = [1, 1];            % initial condition
xk = x0;                % state at the moment
xN = x0;                % iteration history
threshhold = 0.001;      % tolerant error

%% Simulation with constant step size
% iteration
eps = 0.1;
T = [1-eps, -eps; eps, 1];
while (norm(xk) > threshhold)
    xk = (T * xk')';    % iteration step
    xN = [xN; xk];
end
N = size(xN(:,1));      % total iteration steps
% plotting state history
figure;
hold on;
box on;
xlabel('steps');
```

```
ylabel('state');
plot(1:N, xN(:,1), 1:N, xN(:,2));
legend('x_1','x_2');
saveas(gca,'E210_cons_state','pdf');
system('pdfcrop E210_cons_state.pdf E210_cons_state.pdf');
% plotting trajectory
figure;
hold on;
box on;
xlabel('x_1');
ylabel('x_2');
plot(xN(:,1), xN(:,2));
saveas(gca,'E210_cons_tra','pdf');
system('pdfcrop E210_cons_tra.pdf E210_cons_tra.pdf');
% plotting cost function history
J1N = 0.5.*xN(:,1).*xN(:,1) + xN(:,1).*xN(:,2);
J2N = -xN(:,1).*xN(:,2);
figure;
hold on;
box on;
xlabel('steps');
ylabel('cost function');
plot(1:N, J1N, 1:N, J2N);
legend('J_1','J_2');
saveas(gca,'E210_cons_cost','pdf');
system('pdfcrop E210_cons_cost.pdf E210_cons_cost.pdf');

%% Simulation with vanishing step size
xk = x0; xN = x0;
k = 0;
eps = 1;
while (norm(xk) > threshhold)
    T = [1-eps, -eps; eps, 1];
    xk = (T * xk')';    % iteration step
    xN = [xN; xk];
    k = k+1;
    eps = 1 / k;
end
N = size(xN(:,1));      % total iteration steps
% plotting state history
figure;
hold on;
box on;
xlabel('steps');
ylabel('state');
plot(1:N, xN(:,1), 1:N, xN(:,2));
```

```
legend('x_1','x_2');
saveas(gca,'E210_van_state','pdf');
system('pdfcrop E210_van_state.pdf E210_van_state.pdf');
% plotting trajectory
figure;
hold on;
box on;
xlabel('x_1');
ylabel('x_2');
plot(xN(:,1), xN(:,2));
saveas(gca,'E210_van_tra','pdf');
system('pdfcrop E210_van_tra.pdf E210_van_tra.pdf');
% plotting cost function history
J1N = 0.5.*xN(:,1).*xN(:,1) + xN(:,1).*xN(:,2);
J2N = -xN(:,1).*xN(:,2);
figure;
hold on;
box on;
xlabel('steps');
ylabel('cost function');
plot(1:N, J1N, 1:N, J2N);
legend('J_1','J_2');
saveas(gca,'E210_van_cost','pdf');
system('pdfcrop E210_van_cost.pdf E210_van_cost.pdf');
```