Stochastic Signal Processing

Lesson 3 – experimental report 1: requirements

Weize Sun

Results from Last Game – Problem (last year)

- When I run the 5000 trades…
  - This told us:
  - When submitting a program, must test it under the predefined conditions, which is, run 5000 trades before submitting!
  - However, by pressing F9 on the code

```
bar(Return_total)
% bar(ranking_result_total)
xlim([0,41])
```

```
>> bar(Return_total)
% bar(ranking_result_total)
xlim([0,41])
```

We can still see the result

```
未定义函数或变量 'm'。

出错 id5 (line 30)
       if isempty(m)

出错 Run Strategies (line 6)
    Strategies_one_trade(5) = id5(c

出错 main (line 46)
        Strategies_one_trade = Run_
```

Why this? I am confused

```
18 —    if Trade_no<=4700
19 —    [m]=find(member_betray==counterparty);
```

I changed it to:

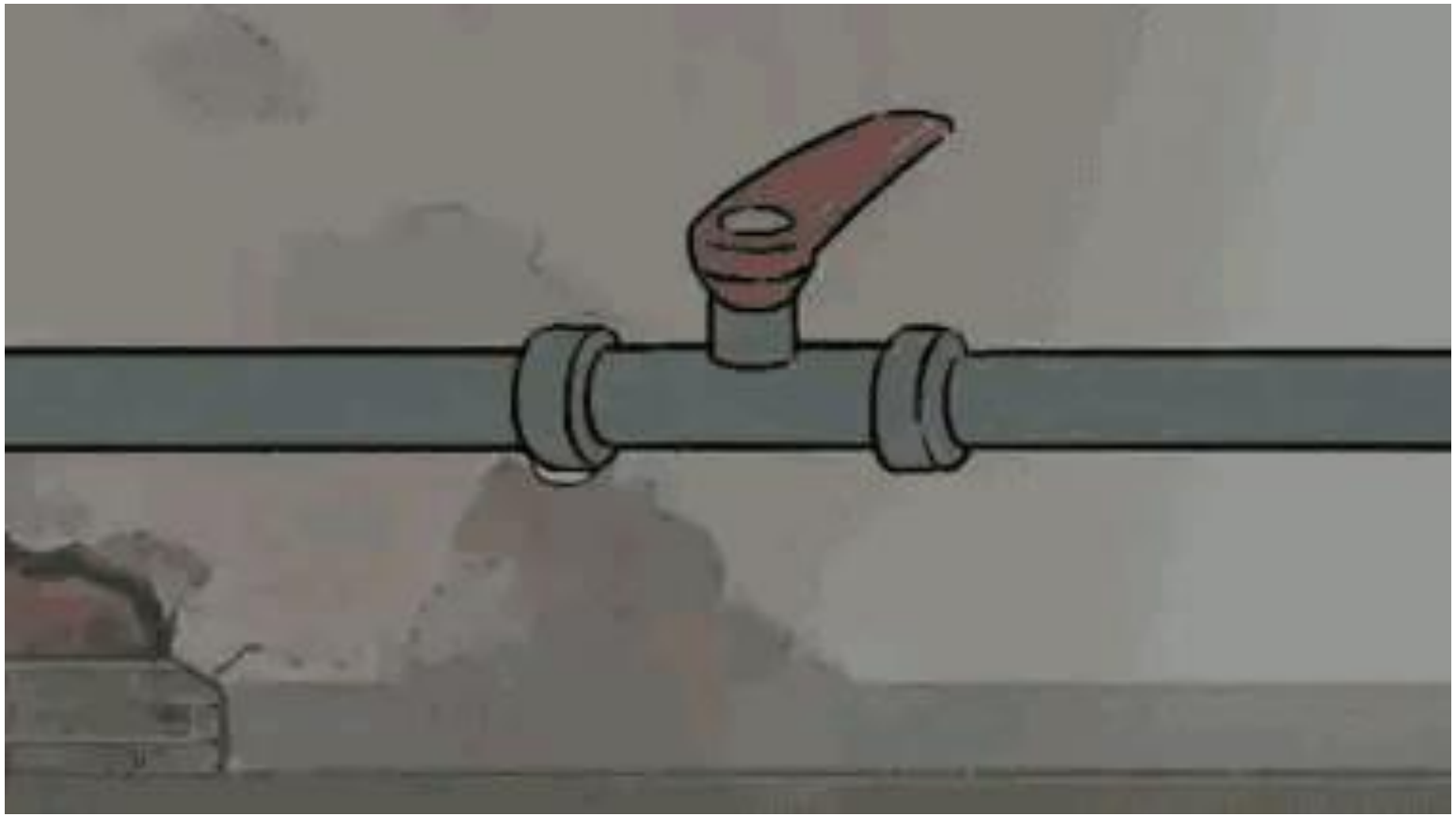
```
-       end
        %     if Trade_no<=4700
-       if 1
-       [m]=find(member_betray==counterparty);
```

Results from Last Game – warning (last year)

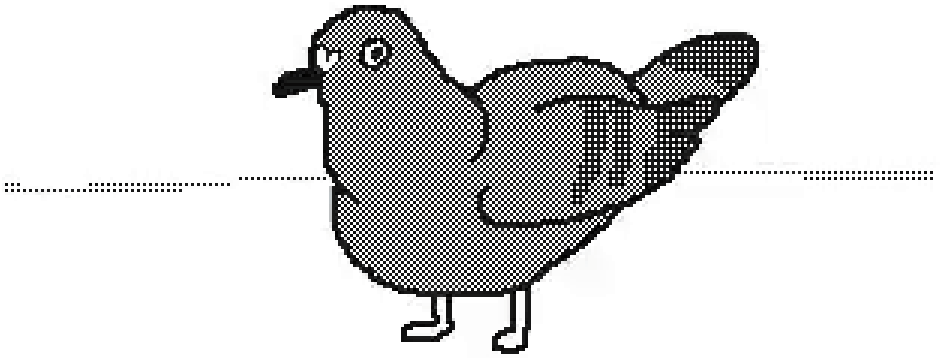
- A programmer Joke:
  - We don't care about warning
  - We care about error only

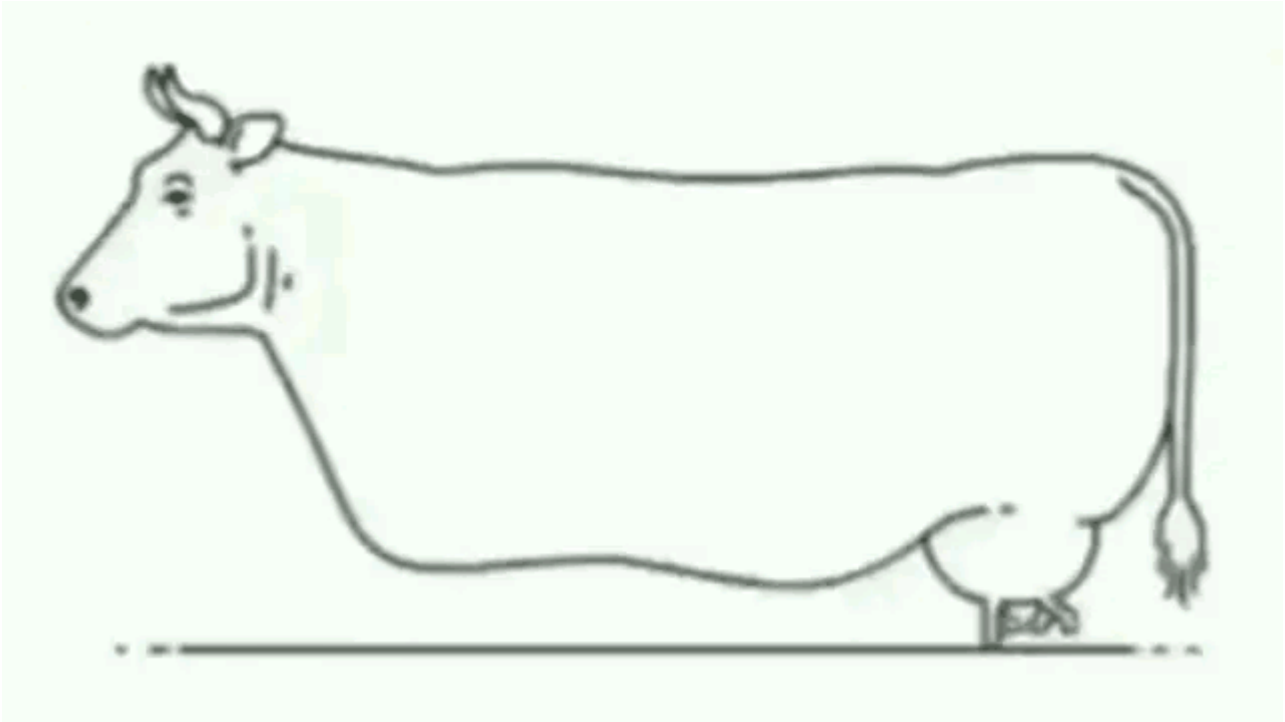
```
警告: 未找到变量 'member_betray'。
> In id5 (line 13)
  In Run Strategies (line 6)
  In main (line 45)
```

```
clear;
clc;
warning off
```



Results from Last Game – warning (some interesting figures)





Results from Last Game – Problem

- Some details

```
12      % Load storage data and information data
13
14 -    load storage_id9.mat Trade_no your_id Trust_no Betray_no;
15 -    load infor_id9.mat counterparty_action;
16
```

```
load storage_id9.mat;
load infor_id9.mat;
```

```
function [your_strategy] = id25(counterparty_id)
    load storage_id25.mat
```

```
function [your_strategy] = id25(counterparty
    load storage_id25.mat
    load infor_id25.mat
```

```
Determine the current phase: the first two transactions betray, the next two reject, the last two trade
phase = mod(Trade_no, 6);
```

```
% Determine the current phase: the first two transactions betray, the next two reject, the last two trade
```

Results from Last Game – something interesting (last year)

- As we know that, we now include 10 NPCs (last year) and
  - Id 31-35 will always betray
  - Id 36-40 will trust if not get betrayed, otherwise reject
- therefore some students include such codes:

```
if counterparty_id>35%
```

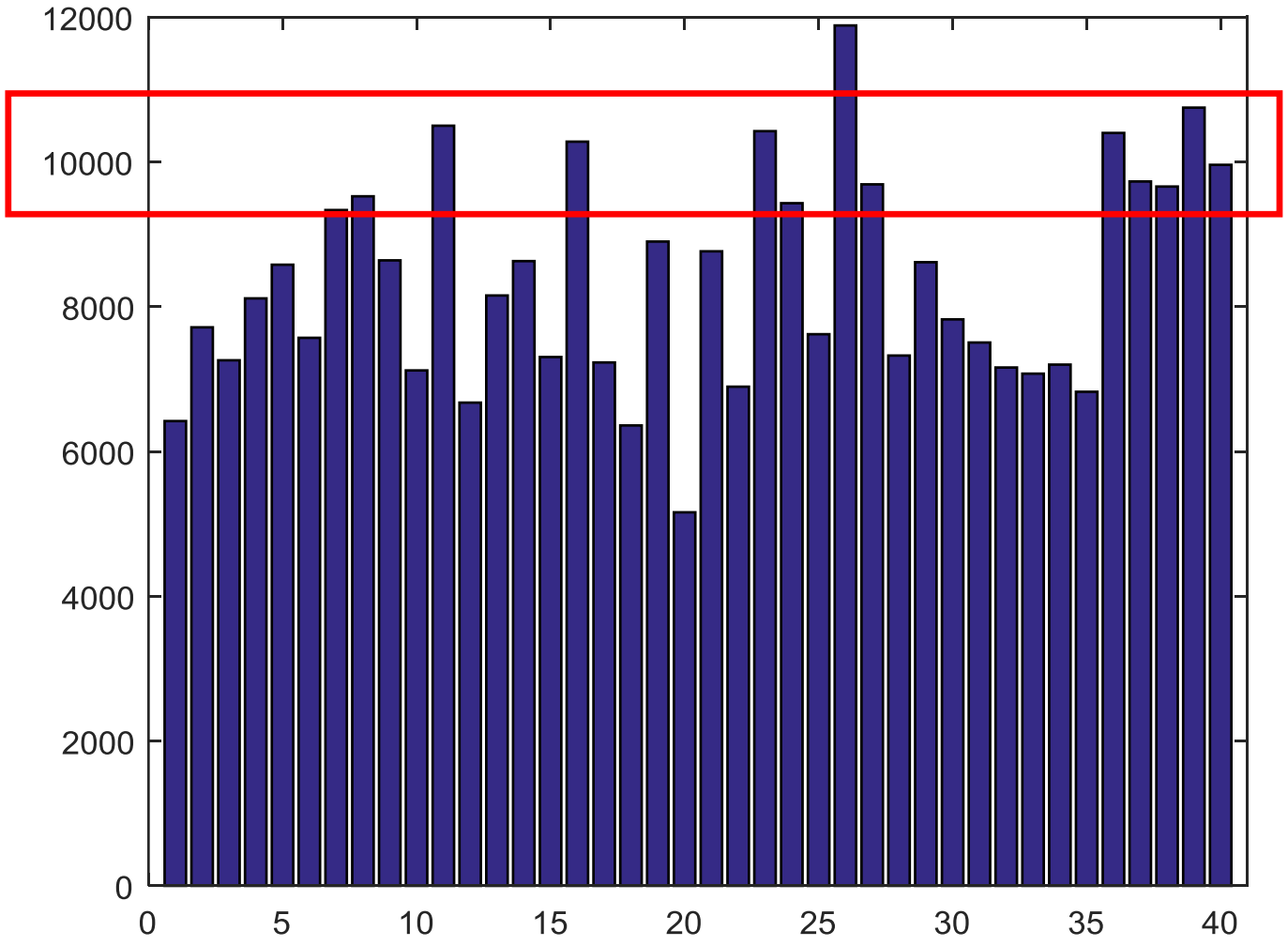
```
if counterparty_now == (31|32|33|34|35)
    your_strategy = -1;
elseif  counterparty_now == (36|37|38|39|40)
    your_strategy = 0;
```

- That is good, that is the na ïve Bayesian

Results from Last Game

- Run 2000 trades (X=Y=5)

```
[ValueList, IDList] = sort (Return_total, 'descend');
```



Variables - IDLi

IDList

40x1 double

	1
1	26
2	39
3	11
4	23
5	36
6	16
7	40
8	37
9	27
10	38
11	8
12	24
13	7
14	19
15	21
16	9

The difference is small, seems due to luck

A good example

```
% Print your student ID and Name here, for example
% 2021280508    Xuanzhe Wu
%%
% your_strategy returns your strategy of the trade this time
% your_strategy = 0 means that you want to trust the counterparty
% your_strategy not equal to 0 means that you want to betray the
% counterparty this time
%%
% counterparty_id is the ID of the counterparty you are going to tr
% this time
%% Now we begins
function [your_strategy] = id26(counterparty_id)
  counterparty_now = counterparty_id;
   persistent z;

   if  ~exist('storage_id26.mat','file')
      your_id = 26;
      Trade_no=0;
      save('storage_id26.mat','your_id','Trade_no')
   else
      load storage_id26.mat
   end

   if  ~exist('infor_id26.mat','file')
      counterparty_id = 0;
      counterparty_action=0;
      save('infor_id26.mat','counterparty_action','counterparty_id')
   else
       load infor_id26.mat
   end
   %Detect the existence of two mat files. If not, create a file.
```

```
if Trade_no~=0 && isempty(z)
    Trade_no=0;
    z=1;
end

if Trade_no==0
    list_betray = [];
    list_trust = [];
end
if counterparty_action > 0
    list_betray = [list_betray; counterparty_id];
elseif counterparty_action == 0
    list_trust = [list_trust; counterparty_id];
end
[m]=find(list_betray==counterparty_now);
[n]=find(list_trust==counterparty_now);
% if m is 'empty 0*0 double', then the counterparty_now
% betrayed you; otherwise, if n is 'empty 0*0 double', th
%trust you.

if isempty(m) && isempty(m)
    your_strategy = 0;     %first trade,I trust.
elseif isempty(m)
    your_strategy = 0;     %never betrayed me,I trust.
elseif isempty(n)
    your_strategy = -1;    %if not,never trust me,I reject.
else
    your_strategy = 1;     %both,I betray.
end
```

```
    Trade_no = Trade_no + 1;
    save storage_id26.mat Trade_no your_id list_betray list_trust
    % ONLY save your data in the file storage_id26.mat,
    % otherwise you will be treated as 'homework not submitted'
end
```

More good examples

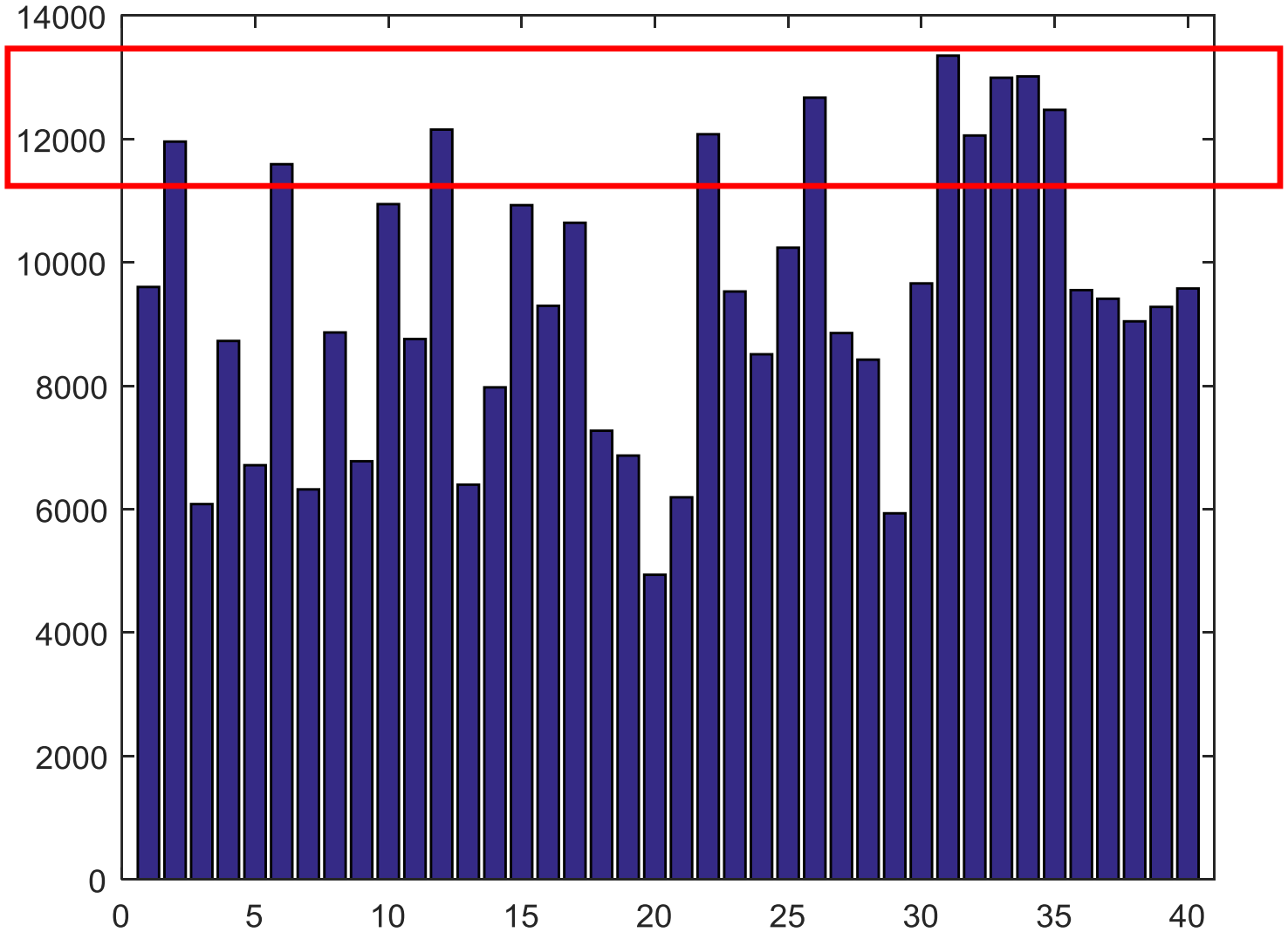
```
% Print your student ID and Name here, for example
% 2021280221    Runlin Feng
%%
% your_strategy: returns your strategy of the trade this time
% your_strategy = 0: means that you want to trust the counterpar
% your_strategy > 0: means that you want to betray the counterp
% your_strategy is others: means that you want to reject the cou
% this time
%%
% counterparty_id: is the ID of the counterparty you are going to
% this time
function [your_strategy] = id11(counterparty_id)
    % we store the counterparty_id from the input to counterparty_
    counterparty_now = counterparty_id;
    % We define a static variable for later use
    persistent flag;
    % your_id: Own serial number
    your_id = 11;

    % Verify that the file exists
    if ~exist('infor_id11.mat','file')
```

```
% Print your student ID and Name here, for example
% 2021280473    Zekun Wu
%%
% your_strategy returns your strategy of the trade this time
% your_strategy = 0 means that you want to trust the counterparty this time
% your_strategy not equal to 0 means that you want to betray the
% counterparty this time
%%
% counterparty_id is the ID of the counterparty you are going to trade with
% this time
%% Now we begins
function [your_strategy] = id23(counterparty_id)
    counterparty_now = counterparty_id;
    load infor_id23.mat
    load storage_id23.mat
    if Trade_no==0
        list_betray = [];
    end
    if counterparty_action > 0
        list_betray = [list_betray; counterparty_id];
    end
```

- Algorithm: good strategy
- System design: good programming
- better: both

Results from Last Game

- Run 2000 trades (X=8, Y=5)



IDList

40x1 double

	1
1	31
2	34
3	33
4	26
5	35
6	12
7	22
8	32
9	2
10	6
11	10
12	15
13	17
14	25
15	30

The Experimental Report 1

The experimental Report 1 contains 3 parts:

- Basic 1 (30 points) : submitted the program in weeks 1 and 2 and it is done.
- Basic 2 (40 points) : a direct change of some program and figuring, because of time this week, it will be introduced next week.
- Advance (30 points) & extra (+10 points): the new games with Bayes' theorem

  Scoring criteria:

  - Correctly use the Bayesian rule to determined the strategy 1 for game 1
  - Explain the strategy 1 correctly and clearly
  - Use appropriate system to perform the testing of strategy 1
  - Correctly design the game 2, and correctly use the Bayesian rule to determined the strategy 2 for game 2, and use appropriate system to perform the testing of strategy 2. (+10 points)

The Experimental Report 1

- Advance (30 points) : the new game with Bayes' theorem

  You will now joint this game 1:

  - You will trade with one counterparty; your counterparty's action can be trust or betray
  - You have two strategies: trust and reject, and the return table is:

		A: Your counterparty	
		trust	betray
B: You	trust	A: +10; B: +10	A: +5; B: -5
	reject	A: 0; B: 0	A: 0; B: 0

  - Before trading, you will be given the following information:
    - Your counterparty's probability of betray follows uniform distribution in (0, 1)
    - You will be given the counterparty's previous 10 actions towards other persons, which is a 10*1 vector of {trust, betray}
    - The system and the default strategy will be given, but you are required to design your own strategy, and explain why you design your strategy like this
  - You will trade with this counterparty 100 times, and show your total return
  - You should estimate your counterparty's probability of betray, but, never directly look at your counterparty's actual probability of betray, otherwise, 0 point.

- more details
  1. "You will trade with this counterparty 100 times": your counterparty's probability of betray is fixed in the very beginning for the whole 100 times.
  2. How to fix it? Generate a random value X uniformly distributed in (0, 1)

"Your counterparty's probability of betray follows uniform distribution in (0, 1)"

```
clear;
clc;
warning off
% the above two commands clear all the previous record in the Memory
% counterparty action: 0 trust, others betray
% your action: 0 trust, others call police

N_trades = 100;                          % trade N_trades times
Return_total = 0;                        % the retrun
counterparty_betray_prob = rand(1);      % randomly initial the probability of betray of the
counterparty
% note that it will last for the whole game
counterparty_previous_action_list = rand(10,1);
counterparty_previous_action = double(counterparty_betray_prob > counterparty_previous_action_list);

for n_trade = 1 : N_trades               % looping
    n_trade                              % just to show the trade no for quick check
    Your_Strategy = Your_Strategies(counterparty_previous_action);
```

The Experimental Report 1

- more details
  1. "You will trade with this counterparty 100 times": your counterparty's probability of betray is fixed in the very beginning for the whole 100 times.
  2. How to fix it? Generate a random value X uniformly distributed in (0, 1)
  3. In each trade, generate a random value Y uniform distribution in (0, 1)
  4. If X>Y, then in this trade, the counterparty's action is betray; otherwise, trust.
  5. Repeat 3-4, until finish all 100 trades

```
for n_trade = 1 : N_trades                    % looping
    n_trade                                   % just to show the trade no for quick check
    Your_Strategy = Your_Strategies(counterparty_previous_action);
    % this time, you can pass anything you want into the 'Your_Strategies',
    % except the 'counterparty_betray_prob'
    % you can change the whole system as you wish
    counterparty_action = double(counterparty_betray_prob > rand(1));
    if Your_Strategy==0
```

Y

The Experimental Report 1

- Advance (30 points) : the new game with Bayes' theorem

  Requirement for the testing of your strategy 1 for this game 1: (Score points)

  1. From **a statistical point of view**, how to design a system (which is, modify the default system) to evaluate your strategy?
     - **For example, test your strategy for 500 independent runs**. But your counterparty's probability of betray must be a r.v with uniform distribution in (0, 1) **in every independent run**, which is, you **should neither set it as a constant or a same r.v in all the independent runs**, otherwise, 0 point.

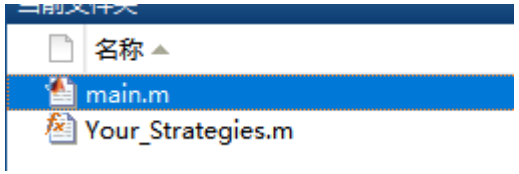
  You should also explain your modification of the default system

  2. Explain the reason of your strategy, must related to probability. (Hint: see example 7, lesson 2)
  3. Show and explain your evaluation result of your strategy.

  Note: You should submit the whole system with your strategy, and your system must be runnable (no error, warning accepted), otherwise, 0 point.

The Experimental Report 1

- The system and the default strategy




```
clear;
clc;
warning off
% the above two commands clear all the previous record in the Memory
% counterparty action: 0 trust, others betray
% your action: 0 trust, others call police

N_trades = 100;                              % trade N_trades times
Return_total = 0;                            % the retrun
counterparty_betray_prob = rand(1);          % randomly initial the p
% note that it will last for the whole game
counterparty_previous_action_list = rand(10,1);
counterparty_previous_action = double(counterparty_betray_prob > counterparty_previous_act

for n_trade = 1 : N_trades                   % looping
    n_trade                                  % just to show the trade no for quick check
    Your_Strategy = Your_Strategies(counterparty_previous_action);
    % this time, you can pass anything you want into the 'Your_Strategies',
    % except the 'counterparty_betray_prob'
    % you can change the whole system as you wish
    counterparty_action = double(counterparty_betray_prob > rand(1));
    if Your_Strategy==0
        if counterparty_action==0
            Return_current = 10;   % both trust, add 10 points
        else
            Return_current = -10;  % self trust, counterparty betray, -10 points
        end
    else
        if counterparty_action==0
            Return_current = -10;   % self call police, counterparty trust, -10 points
```

```
function Your_Strategy = Your_Strategies(counterparty_previous_action)
    % this is only a default strategy, it is not good
    Your_Strategy = double(0.5 > rand(1));
    % as the mean of the betray rate of your counterparty is 0.5, 50% trust
    % and 50% call police
end
```

The Experimental Report 1

- Extra (+10 points): the new game with Bayes' theorem

  You will now joint this game 2:

  - You will trade with one counterparty; your counterparty's action can be trust or betray
  - You have two strategies: trust and reject, and the return table is:

		A: Your counterparty	
		trust	betray
B: You	trust	A: +10; B: +10	A: +5; B: -5
	reject	A: 0; B: 0	A: 0; B: 0

  - Before trading, you will be given the following information:
    - Your counterparty's probability of betray follows uniform distribution in [0.4, 0.8]
    - You have 100 friends, and each of they had already trade with this counterparty 100 times independently, your friends will tell you how many times of 'betray' out of 100 this counterparty did in their trading
  - You will trade with this counterparty 1 time only
  - You will perform the above for **200 independent runs**, in every **independent runs**, you will trade with different counterparty therefore their probability will be different (but all follows uniform distribution in [0.4, 0.8])

The Experimental Report 1

- Extra (+10 points): the new game with Bayes' theorem

  Requirement for the testing of your strategy 2 for this game 2: (Score points)

  1. You should submit the whole system with your strategy, and your system must be runnable (no error, warning accepted), otherwise, 0 point.
     - You should design your testing system all by yourself, your counterparty's probability of betray must be a r.v with uniform distribution in [0.4, 0.8] in every independent run, which is, you **should neither set it as a constant or a same r.v in all the independent runs**, otherwise, 0 point.
  2. You should **estimate your counterparty's probability of betray**, but, never directly look at your counterparty's actual probability of betray, otherwise, 0 point. Then, you can show your **statistical** return, and explain your evaluation result of your strategy.
     - For 'evaluation', there are many evaluating indicator, for example, record the total return, or record the action as 'success or fail'(but you should give the definition of success and fail)
  3. Explain the reason of your strategy, must related to probability. (Hint: see example 8, lesson 3)