ET 601: Computer Applications for Engineers **2013/2**

HW Solution 3 — Due: January 8

Lecturer: Asst. Prof. Dr.Prapun Suksompong (prapun@siit.tu.ac.th)

Instructions

(a) ONE part of a question will be graded (5 pt). Of course, you do not know which part will be selected; so you should work on all of them.

(b) It is important that you try to solve all problems. (5 pt)

(c) Submit your work as one pdf file (which contains the solution for all the questions). The PDF file name should be ET601_HW3_FIRSTNAME.pdf in which the FIRSTNAME part is replaced by your first name.

(d) For question that involved MATLAB (Q1), the solution should contain both the MATLAB codes and the resulting figures. If answers are also displayed in the command window, they should be captured and shown in your solution as well.

(e) For analytical questions (Q2-Q4), write down all the steps that you have done to obtain your answers. You may not get full credit even when your answer is correct without showing how you get your answer.

(f) Late submission will be heavily penalized.

Problem 1. Monty Hall Game

(a) Write a MATLAB script to evaluate and plot the following relative frequencies from the Monty Hall game.

(i) The winning probability when the player always switch to the remaining door at the last step of the game.

(ii) The winning probability when the player always sticks with the original choice at the last step of the game.

Your code should have parts that (automatically) simulate all steps of the game. In particular, it should explicitly randomize the winning doors, the door that the player originally choose, and the goat door that the host chooses to reveal (when he has choices, i.e., when the player's original selection happened to be correct).

(b) Monty Crawl Problem: Implement the following modified version of the game: As in the original game, once the player has selected one of the three doors, the host then reveals one non-selected door which does not contain the car. However, the host is old and always very tired, and crawls from his position (near Door #1) to the door he is to open. In particular, if he has a choice of doors to open (i.e., if the player's original selection happened to be correct), then he opens the smallest number available door. (For example, if the player selected Door #1 and the car was indeed behind Door #1, then the host would always open Door #2, never Door #3.) [Rosenthal, 2005]

A smart player would realize that if the host opens the higher-numbered unselected door, then the car must be in the one remaining (lower-numbered) unselected door. A naive player would not take this fact into consideration.

Evaluate and plot the following relative frequencies:

(i) The winning probability for the naive player who always switches to the remaining door at the last step of the game.

(ii) The winning probability for the naive player who always sticks with the original choice at the last step of the game.

(iii) The winning probability for the smart player who first observes the door number that the host opened. If the host opens the higher-numbered door, this player always switches to the remaining door. If the host opens the lower-numbered door, this player always sticks with the original choice.

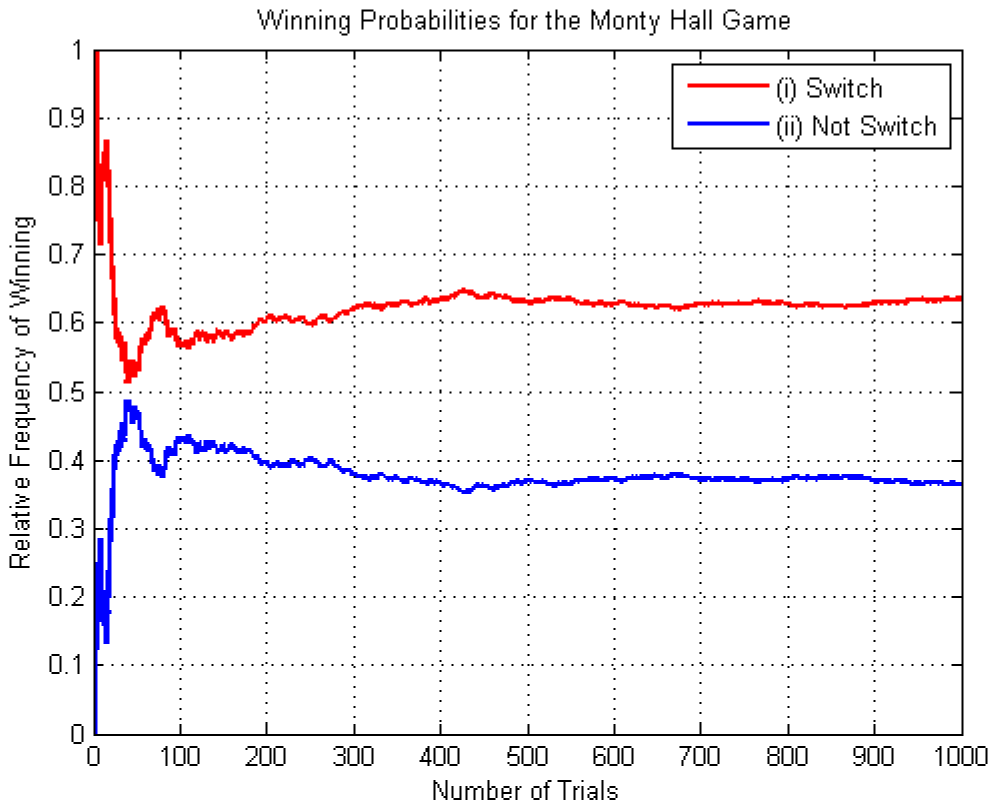
Note that there is also another strategy for the smart player. If the host opens the lower-numbered door, the player may always switch to the remaining door. However, because the smart player always switch to the remaining door if the host opens the higher-numbered door, this strategy is exactly the same as simply switching to the remaining door regardless of what the host did and therefore it is the same as strategy (ii) above.

Solution:

(a) See file `Monty_Sol.m`.

(b) See file `Monty_Crawl_Sol.m`.

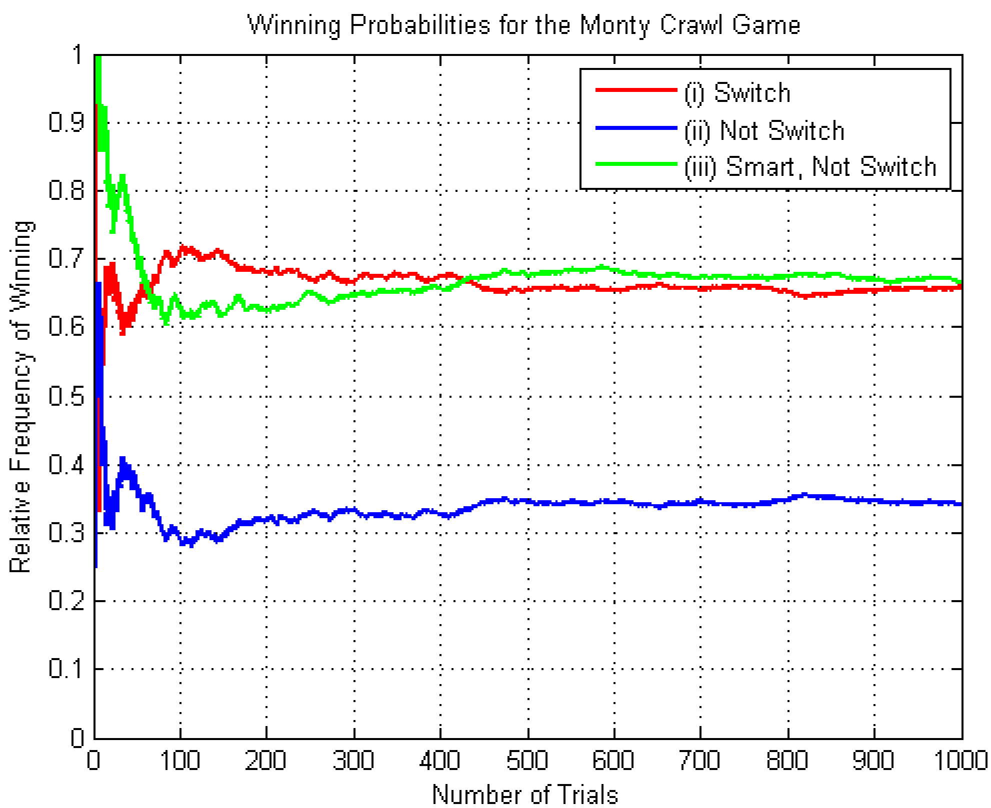
```matlab
% Monty_Sol use simulation to estimate the winning probabilites in the
% Monty Hall game.

clear all; close all;
D = 3; % number of doors
N = 1e3; % #times to repeat the game

% Initialization
car = randi(D,1,N); % The winning door
X1 = randi(D,1,N); % Player's original choice

OriginalCorrect = (car == X1);
% Preallocation
X_H = zeros(1,N); % Door open by the Host
X_S = zeros(1,N); % Remaining door that the player can switch to
X_NS = X1;
for k=1:N
    OC = OriginalCorrect(k);
    % Host's selection
    % A is a temporary variable to keep the choice(s) under consideration
    A = 1:D; % Start with all the choices for doors
    A = A(A~=X1(k)); % Remove the original choice
    A = A(A~=car(k)); % Remove the car position
    if OC % If original choice of the player was correct,...
        X_H(k) = A(randi(2)); % Host randomly chooses one of the two remaining doors
    else % If orginal choice of the player was incorrect,...
        X_H(k) = A; % Host chooses the one remaining door
    end
    % Player's selection
    A = 1:D; % Start with all the choices for doors
    % X_NS(k) = X1(k); % This line is not needed because we have already
    % equated X_NS and X1 earlier.
    A = A(A~=X1(k)); % Remove the original choice
    A = A(A~=X_H(k)); % Remove the host choice
    X_S(k) = A; % Swicth to the one remaining door
end

SwitchCorrect = (car == X_S);
p_switch = cumsum(SwitchCorrect)./(1:N);
plot(1:N,p_switch,'r','LineWidth',1.5)

hold on

p_original = cumsum(OriginalCorrect)./(1:N);
plot(1:N,p_original,'LineWidth',1.5)

legend('(i) Switch','(ii) Not Switch')

ylabel('Relative Frequency of Winning')
xlabel('Number of Trials')
grid on
title('Winning Probabilities for the Monty Hall Game')
```



Published with MATLAB® R2013a

```
% Monty_Crawl_Sol computes the winning probabilities in the Monty Crawl
% game. Three player strategies are considered.

clear all; close all;
D = 3; % number of doors
N = 1e3; % #times to repeat the game

% Initialization
car = randi(D,1,N); % The winning door
X1 = randi(D,1,N); % Player's original choice

OriginalCorrect = (car == X1);
% Preallocation
X_H = zeros(1,N); % Door open by the Host
X_S = zeros(1,N); % Remaining door that the player can switch to
X_Smart_S = zeros(1,N);
X_Smart_NS = zeros(1,N);
H_ChooseL1 = zeros(1,N);
X_NS = X1;
for k=1:N
    OC = OriginalCorrect(k);
    % Host's selection
    % A is a temporary variable to keep the choice(s) under consideration
    A = 1:D; % Start with all the choices for doors
    A = A(A~=X1(k)); % Remove the original choice
    A = A(A~=car(k)); % Remove the car position
    if OC % If orginal choice of the player was correct,...
        X_H(k) = A(1); % Host always chooses the lower number door that is left
    else % If orginal choice of the player was incorrect,...
        X_H(k) = A; % Host chooses the one remaining door
    end
    % Player's selection
    A = 1:D; % Start with all the choices for doors
    % X_NS(k) = X1(k); % This line is not needed because we have already
    % equated X_NS and X1 earlier.
    A = A(A~=X1(k)); % Remove Player's original choice
    if X_H(k)==A(2) % Host choose the higher number. We then know that the car must be in the
lower number.
        % H_ChooseL1(k) = 0; % not needed
        X_Smart_S(k) = A(1);
        X_Smart_NS(k) = A(1);
        X_S(k) = A(1);
    else
        H_ChooseL1(k) = 1;
        A = A(A~=X_H(k)); % Remove the host choice
        X_Smart_NS(k) = X1(k);
        X_Smart_S(k) = A; % Switch to the one remaining door
        X_S(k) = A;
    end
end

SwitchCorrect = (car == X_S);
```

```
p_switch = cumsum(SwitchCorrect)./(1:N);
plot(1:N,p_switch,'r','LineWidth',1.5)
hold on

p_original = cumsum(OriginalCorrect)./(1:N);
plot(1:N,p_original,'LineWidth',1.5)

SmartNotSwitchCorrect = (car == X_Smart_NS);
p_smartnotswitch = cumsum(SmartNotSwitchCorrect)./(1:N);
plot(1:N,p_smartnotswitch,'g','LineWidth',1.5)

legend('(i) Switch','(ii) Not Switch','(iii) Smart, Not Switch')
ylabel('Relative Frequency of Winning')
xlabel('Number of Trials')
grid on
title('Winning Probabilities for the Monty Crawl Game')
```



Published with MATLAB® R2013a

Problem 2. If A, B, and C are disjoint events with $P(A) = 0.2$, $P(B) = 0.3$ and $P(C) = 0.4$, determine the following probabilities:

(a) $P(A \cup B \cup C)$

(b) $P(A \cap B \cap C)$

(c) $P(A \cap B)$

(d) $P((A \cup B) \cap C)$

(e) $P(A^c \cap B^c \cap C^c)$

[Montgomery and Runger, 2010, Q2-75]

Solution:

(a) Because A, B, and C are disjoint, $P(A \cup B \cup C) = P(A) + P(B) + P(C) = 0.3 + 0.2 + 0.4 = \boxed{0.9.}$

(b) Because A, B, and C are disjoint, $A \cap B \cap C = \emptyset$ and hence $P(A \cap B \cap C) = P(\emptyset) = \boxed{0}$.

(c) Because A and B are disjoint, $A \cap B = \emptyset$ and hence $P(A \cap B) = P(\emptyset) = \boxed{0}$.

(d) $(A \cup B) \cap C = (A \cap C) \cup (B \cap C)$. By the disjointness among A, B, and C, we have $(A \cap C) \cup (B \cap C) = \emptyset \cup \emptyset = \emptyset$. Therefore, $P((A \cup B) \cap C) = P(\emptyset) = \boxed{0}$.

(e) From $A^c \cap B^c \cap C^c = (A \cup B \cup C)^c$, we have $P(A^c \cap B^c \cap C^c) = 1 - P(A \cup B \cup C) = 1 - 0.9 = \boxed{0.1.}$

Problem 3. The sample space of a random experiment is $\{a, b, c, d, e\}$ with probabilities 0.1, 0.1, 0.2, 0.4, and 0.2, respectively. Let A denote the event $\{a, b, c\}$, and let B denote the event $\{c, d, e\}$. Determine the following:

(a) $P(A)$

(b) $P(B)$

(c) $P(A^c)$

(d) $P(A \cup B)$

(e) $P(A \cap B)$

[Montgomery and Runger, 2010, Q2-55]

Solution:

(a) Recall that the probability of a finite or countable event equals the sum of the probabilities of the outcomes in the event. Therefore,

$$\begin{aligned} P(A) &= P(\{a, b, c\}) = P(\{a\}) + P(\{b\}) + P(\{c\}) \\ &= 0.1 + 0.1 + 0.2 = \boxed{0.4.} \end{aligned}$$

(b) Again, the probability of a finite or countable event equals the sum of the probabilities of the outcomes in the event. Thus,

$$\begin{aligned} P(B) &= P(\{c, d, e\}) = P(\{c\}) + P(\{d\}) + P(\{e\}) \\ &= 0.2 + 0.4 + 0.2 = \boxed{0.8.} \end{aligned}$$

(c) $P(A^c) = 1 - P(A) = 1 - 0.4 = \boxed{0.6.}$

(d) Note that $A \cup B = \Omega$. Hence, $P(A \cup B) = P(\Omega) = \boxed{1.}$

(e) $P(A \cap B) = P(\{c\}) = \boxed{0.2.}$

Problem 4. Let A and B be events for which $P(A)$, $P(B)$, and $P(A \cup B)$ are known. Express the following probabilities in terms of the three known probabilities above.

(a) $P(A \cap B)$

(b) $P(A \cap B^c)$

(c) $P(B \cup (A \cap B^c))$

(d) $P(A^c \cap B^c)$

Solution:

(a) $P(A \cap B) = \boxed{P(A) + P(B) - P(A \cup B)}$. This property is shown in class.

(b) We have seen in class that $P(A \cap B^c) = P(A) - P(A \cap B)$. Plugging in the expression for $P(A \cap B)$ from the previous part, we have

$$P(A \cap B^c) = P(A) - (P(A) + P(B) - P(A \cup B)) = \boxed{P(A \cup B) - P(B)}.$$

Alternatively, we can start from scratch with the set identity $A \cup B = B \cup (A \cap B^c)$ whose union is a disjoint union. Hence,

$$P(A \cup B) = P(B) + P(A \cap B^c).$$

Moving $P(B)$ to the LHS finishes the proof.

(c) $P(B \cup (A \cap B^c)) = \boxed{P(A \cup B)}$ because $A \cup B = B \cup (A \cap B^c)$.

(d) $P(A^c \cap B^c) = \boxed{1 - P(A \cup B)}$ because $A^c \cap B^c = (A \cup B)^c$.