Monty Hall: Switch?

Elijah Dunn

05/04/2016

Introduction

The Monty Hall problem is a classic exploration of the divergence from intuition that probability and statistics can bring. Neglecting problems associated with the host not being completely random, the analysis can proceed one of two ways: simple list and formal conditional probability assessment. In addition, simple numerical simulation can approximate the solution with random inputs.

List

Since we are dealing with only three doors, a list of possible outcomes is relatively short. Furthermore, the doors are identical and therefore choosing "Door 1" every time is equivalent to choosing any door.

Table 1: Outcomes of possible Monty Hall games with switching and staying. (Adapted from [1])

Game	Door 1	Door 2	Door 3	Result
1	Auto	Goat	Goat	Lose if switch
2	Goat	Auto	Goat	Win if switch
3	Goat	Goat	Auto	Win if switch
4	Auto	Goat	Goat	Win if stay
5	Goat	Auto	Goat	Lose if stay
6	Goat	Goat	Auto	Lose if stay

As Table 1 indicates, switching doors results in a 2/3 chance of winning while staying results in 1/3 chance. Switching doors doubles the chance of winning (compared to staying).

Conditional

For the more formal analysis with conditional probabilities relies on the additional information given by opening an unchosen door. Since the original decision was made under random circumstances, it's probability is 1/3. That leaves 2/3 probability distributed between the remaining two doors (1/3 each). Once the goat has been revealed, the two remaining door sample space is collapsed to the one remaining door giving it the entire probability of 2/3. The contestants choices are therefore 1/3 for staying and 2/3 for switching.

Simulation

Famously, Paul Erdős refused to accept the solution until he was shown a computer simulation. Source code for the simulation provided is attached in Appendix A.

```
./monty

For 1.0e+07 games played:
Switching resulted in 6666551 wins (~66.666%)
Staying resulted in 3333880 wins (~33.339%)
```

Fig 1: Running the simulation with 10 million trials.

Figure 1 shows an approaching probability of 1/3 for staying and 2/3 for switching.

Conclusion

Three separate analyses of the Monty Hall problem indicate that switching doors will likely double your chance of winning. More analysis is necessary to include the possibility that Monty's choice is non-random.

References

1. "Game Show Problem | Marilyn Vos Savant." 2016. Accessed Apr 20. http://marilynvossavant.com/game-show-problem/.

Appendix A

```cpp
#include <iostream>
#include <ctime>
#include <cstdlib>
#include <iomanip>
#include <math.h>

//global variables
const int num_games = 10000000;

//Function: get a random integer between 1 and max (inclusive)
int getrand(int max){
  return (max*double(std::rand()))/(RAND_MAX)+1;
}

//returns the door number that is neither door_1 nor door_2
int other_door(int door_1, int door_2){
  if (door_1 == door_2) //check for quantum indeterminacy
    return 0;
  switch(door_1+door_2){ //simplified Look Up Table
    case 5:
      return 1;
    case 4:
      return 2;
    case 3:
      return 3;
    default:
      return 0; //error
  }
```

```
}

//returns the door number that monty opens
int monty(int winner, int choice){
  //if player chose the winning door
  if(winner == choice)
    return (winner + getrand(2) - 1) % 3 + 1; //reveal other door at random
  //else player chose a loosing door
  return other_door(winner, choice);
}

/* Function: play the game
* Car is stored behind a door at random.
* Player picks door at random.
* Monty picks a door to discard.
* If stay == false, switch player choice.
* Return true if player picked the correct door.
*/
bool play(bool stay){
  int const winning_door = getrand(3); //car is placed behind a random door
  int player_choice = getrand(3); //player chooses a random door

//monty opens a door
  int open_door = monty(winning_door, player_choice);

//player switches doors if not staying with current choice
  if (!stay){
    player_choice = other_door(player_choice, open_door);
  }

  if (winning_door == player_choice)
    return true;

  return false;
}


int main(){

  //initialize random number generator srand
  std::srand(std::time(NULL));

  /* Simple Solution
  * Assume Monty Hall chooses the door to be discarded at random.
  * Assume switching must be defined before playing.
  */

  //play the game (call)

  //never switch doors
  int switch_wins = 0;
  for(int i = 0; i < num_games; i++){
    bool win = play(false);
```

```
    if(win)
      switch_wins++;
  }

  //always stay
  int stay_wins = 0;
  for(int i = 0; i < num_games; i++){
    bool win = play(true);
    if(win)
      stay_wins++;
  }

  //calculate percent wins
  double per_switch, per_stay = 0.0;
  per_switch = double(switch_wins)/num_games*100;
  per_stay = double(stay_wins)/num_games*100;

  //print comparison
  std::cout << std::scientific << std::setprecision(1);
  std::cout << std::endl << "For " << double(num_games) << " games played:" << std::endl;
  std::cout << std::fixed << std::setprecision(3);
  std::cout << "Switching resulted in " << switch_wins << " wins ";
  std::cout << "(~" << per_switch << "%)" << std::endl;
  std::cout << "Staying resulted in " << stay_wins << " wins ";
  std::cout << "(~" << per_stay << "%)" << std::endl << std::endl;

  return 0;
}
```