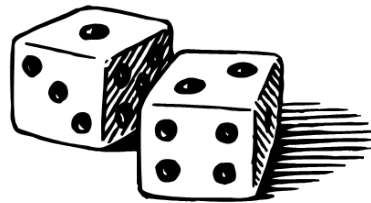


**Mini-project:
Five of a kind in Yatzy**



How many dices should one throw in order get Yatzy? This and other questions are answered in this project. You will work with some **probability, statistics and linear algebra** during this project.

March 06, 2012

1 Assignment

Five dice is all that is needed to play the old dice game Yatzy, in which the participants take turn to throw the dice to obtain certain combinations. The highest score is obtained for Yatzy, *i.e.* five of a kind. When it is his or her turn each participant is allowed to throw the dice three times. After each throw the participant is allowed to save, or leave out, as many dice from the next throw as he or she sees fit to increase the chance of a good combination.

Here we ask a related question: How large is the probability to get five of a kind in a given number of throws (not limiting the number of throws to three)? What is the expected value (see definition in Section 3) of throws needed to get five of a kind? A measure of the spread (how common it is to be close to the mean) is the variance, also defined in Section 3.

1.1 Examination

- Produce a MATLAB function that simulates the situation above, and computes and returns the expected value and variance for the number of throws needed to get five of a kind. Draw a histogram for the number of throws needed and compare to the analytical solution found in Section 3.
- If you are interested: How does the solution change if the dice used are loaded, that is if the probability to get six for instance is p , $0 \leq p \leq 1$? What if you have more than 5 dice? What if you have more than 6 sides on the dice?

2 Suggested Solution Outline

First read though the problem specification and the suggested solution outline and try to understand how it divides the problem into smaller subproblems, each of which is easier to solve than the original problem.

The steps below provide one way to split the main programming problem into logical subproblems. The subproblems are deliberately chosen so that it is possible to test the code before continuing with the next step. Do take these opportunities! Following the suggestions closely can result in many small functions solving small parts of the problem, however, it may not be the best, most efficient, nor smartest way to call these small functions to solve the whole problem. Some of the smaller functions (all?) are better suited to be pasted into the code where needed.

Note that this solution outline is intended for students with no or little previous programming experience. A skilled programmer would probably, based on previous experience, divide the problem into different and/or fewer parts. If you consider yourself a skilled programmer and think that you have a better

solution to the problem you are allowed to solve it your way. However, if you need help, the teaching assistant has more experience of the outlined solution.

2.1 Several Throws

Write a function to simulate a throw with a given number of dice and returns the result. Let the number of dice be an argument to the function and return the result as a vector with the results of the throw. Hint: Create a vector with random numbers in the interval 0 to 1, one number for each die, multiply by 6, and round upwards.

Make sure that the probability for a dice to show 1, 2, 3, 4, 5, or 6 is identical. One way to do this is to throw many dice, *e.g.*, 1000, and plot the histogram of the outcome (see `help hist`). How is the histogram supposed to look?

2.2 Count Number of Each Outcome

Write a function taking a set of dice and returns a vector with the number of ones, twos, and so on. Ask the teaching assistant for help if you get stuck here for too long!

Example:

- The outcome [1 4 2 2 4] should yield [1 2 0 2 0 0] (one one, two twos, no threes, two fours, and no fives or sixes).

2.3 Which Outcome is Most Common

Modify the function from the last step to return the most common outcome (ones, twos, threes, ...). The vector previously returned should come handy. If there are two pairs in the result, just return one of them. Test the function and make sure it works properly!

Example:

- The outcome [4 5 4 4 1] should yield 4 (four is the most common outcome).
- The outcome [1 4 2 2 4] should yield either 2 or 4.

2.4 Find the Dice to Throw Again

With the knowledge of the most common outcome it is time to decide which dice to throw again. There are two ways to do this:

1. One alternative is to modify the function from the last step so that it returns a vector containing the indices of the dice to throw again, *i.e.*, the dice not showing the number found in step 2.3. Test the function!

- The dice [4 5 4 4 1] should yield [2 5] (throw dice number 2 and 5 again).
 - The dice [1 4 2 2 4] should yield [1 2 5] or [1 3 4].
2. The second alternative is to put the dice to save at the beginning of the dice vector. Since the value and number of dice is known (Sections 2.3 and 2.2) it is possible to create a vector with the correct number of dice and number. After modifying the function make sure to test it before you proceed!

Example:

- The outcome [4 5 4 4 1] should yield [4 4 4 * *] (where * will be thrown again and so the value is unimportant).
- The outcome [1 4 2 2 4] should yield either [2 2 * * *] or [4 4 * * *].

2.5 Five of a Kind

Now extend the function to throw the selected dice again, and to repeat this procedure until five of a kind is obtained (*i.e.*, there are no dice to throw again). Let the function return how many throws were needed.

Before continuing with the next step make sure the function works. One way to do this is to, just for now, display and study the result after each throw.

Be careful with the situation when you first get 2 of a kind on one throw and on the next throw get 3 of a kind, then you should save the one with 3 of a kind instead of the one you saved at first.

2.6 Monte Carlo Simulation

To use Monte Carlo simulations is to perform an experiment many times in order to get an idea of the underlying probability function. Write a (new) function that throws dice to decide how many throws are needed to get five of a kind several times. Store the number of throws needed in each experiment in a vector. Plot a histogram to illustrate the result (*i.e.*, the number of throws needed). The histogram bins should have width 1. The Monte Carlo function should take the number of experiments as an input argument.

2.7 Estimate Expected Value and Variance

Modify the function to also return the estimated expected value and variance for the throws needed. The mean of throws needed in the experiments yields estimate of the expected value, and a formula to estimate the variance is given in Section 3. In this case x_i is the number of throws needed in experiment i , and n is the number of experiments. Ask the teaching assistant if you cannot work out how to use the formulas in a few minutes!

Section 3 also presents the theoretical expected value and variance. Make sure the estimates from your function are close to the exact values!

2.8 Final examination

Finally, extend your function to also plot the analytical probability function in the same figure as the histogram from the Monte Carlo simulations. The analytical probability function can be found in Section 3. To compare the results keep the following in mind: The sum of the heights of the bars in the histogram equals the number of experiments in the Monte Carlo simulation. On the other hand the sum of the probabilities in the probability function adds up to 1 (five of a kind will turn up sooner or later). Hence, to get comparable results either the values in the histogram or the probability function must be scaled.

If you called your function `yatzzy` the following command in MATLAB should produce a plot with the estimated probability function (the histogram) and the analytic solution, as well as the estimated mean and variance.

```
>> [xhat, s2hat]= yatzzy(experiments)
```

3 Useful Facts

Probability Theory: Definitions

Let X denote the outcome (result) of a die throw (X is a so called stochastic variable). P denote the probability for a certain outcome, e.g., $P(X = 3)$ is the probability to get a three.

The probability function $p_X(k)$ is defined to be $p_X(k) = P(X = k)$.

To throw several dice, or the same die several times, are independent events. It follows that $P(X_1 = k_1, X_2 = k_2) = P(X_1 = k_1) \cdot P(X_2 = k_2)$.

For a fair die the probabilities of the outcomes are $p_X(k) = \frac{1}{6}$, for $k = 1, \dots, 6$.

Another example of a stochastic variable; let Y be the number of throws needed to get five of a kind.

Expected value (mean) for a stochastic variable Y can be interpreted as the average outcome of an experiment ("How many throws are needed on average to get five of a kind?").

The variance is a spread measure that tells how large the average (squared) distance to the mean is. If the variance is small that implies that the number of throws needed to get five of a kind stays approximately the same from time to time, whereas a large variance indicate that the number of throws may vary substantially.

The mean can be estimated using

$$\hat{m} = \frac{1}{n} \sum_{i=1}^n x_i$$

and the variance with

$$\hat{\sigma}^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \hat{m})^2.$$

3.1 The Probability Density Function in the Assignment

It turns out that the probability function in the assignment can be described using a matrix: Let $p(k)$ be the probability that k throws are needed to get five of a kind. It then holds that

$$p(k) = e_1^T A^k e_5$$

where

$$A = \begin{pmatrix} 0 & \frac{1}{6} & \frac{1}{36} & \frac{1}{216} & \frac{1}{1296} \\ 0 & \frac{5}{6} & \frac{10}{36} & \frac{216}{15} & \frac{1296}{25} \\ 0 & 0 & \frac{25}{36} & \frac{216}{80} & \frac{1296}{250} \\ 0 & 0 & 0 & \frac{120}{216} & \frac{900}{1296} \\ 0 & 0 & 0 & 0 & \frac{120}{1296} \end{pmatrix}, \quad e_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad e_5 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

The expected value of the number of throws needed is $\frac{191283}{17248} \approx 11.0902$, and the variance is $\frac{12125651655}{297493504} \approx 40.7594$.

For the interested: The above formulas are derived using the following approach: The probability to get five of a kind in k throws is equal to the probability to have four of a kind in throw $k-1$ and in throw k get another of the same kind, together with the probability of three of a kind in throw $k-1$ and get another two in throw k , etc. The vector $x(k)$ can now be constructed,

$$x(k) = \begin{pmatrix} P(\text{first time with five of a kind in throw } k) \\ P(\text{four of a kind in throw } k) \\ P(\text{three of a kind in throw } k) \\ P(\text{two of a kind in throw } k) \\ P(\text{all different in throw } k) \end{pmatrix}$$

and the probabilities be calculate recursively as $x(k) = Ax(k-1) = A^2x(k-2) = \dots$, where A contains the probabilities for the result in one throw.