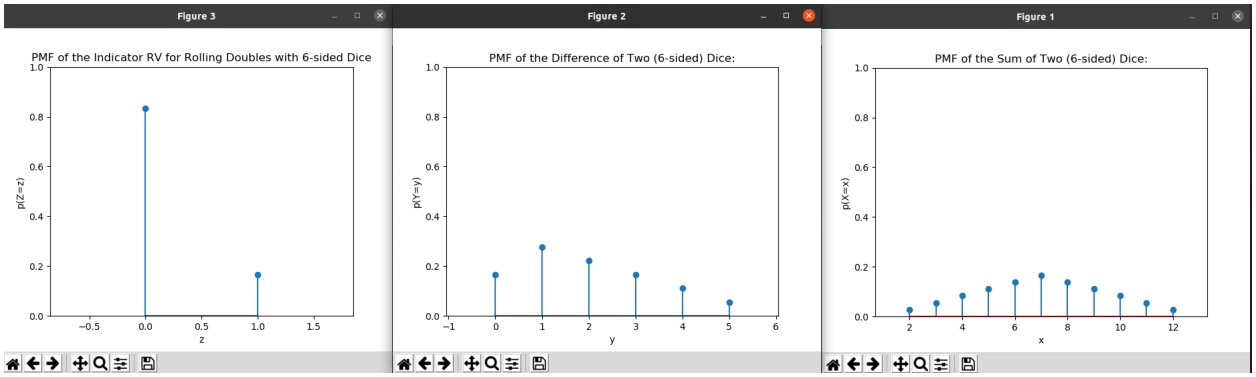
Coding Set 1 Write-up

Section 1 Output



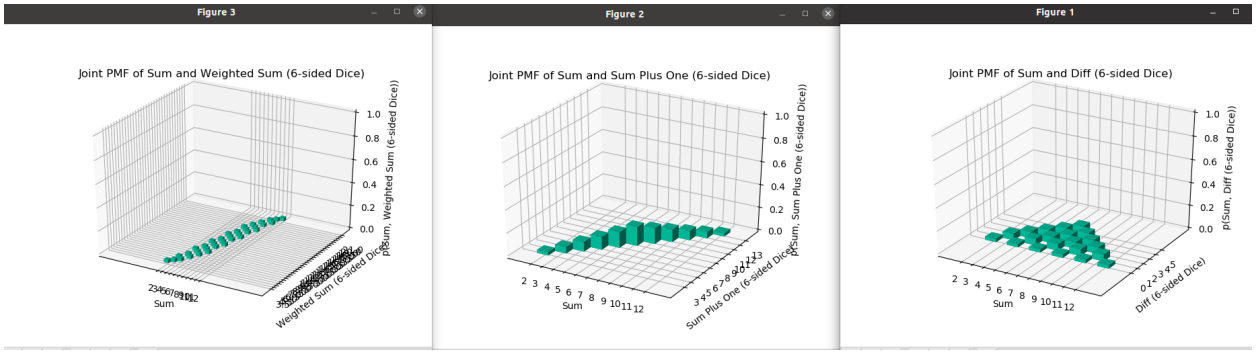
Expected Value of the Sum of Two (6-sided) Dice: 7.0
Variance of the Sum of Two (6-sided) Dice: 5.833333333333333
Expected Value of the Diff. of Two (6-sided) Dice: 1.9444444444444446
Variance of the Difference of Two (6-sided) Dice: 2.052469135802469
Expected Value of the Indicator RV for Rolling Doubles (6-sided Dice): 0.16666666666666669

```
daniel@CTB455-Salmon:~/EC EN 633/CodingSet1/lab1-prob-review$ pytest -v -rN --tb=no --no-header test/test_part1.py
============================= test session starts ==============================
collected 15 items

test/test_part1.py::TestFunctionEvaluatePMF::test_throws_when_probabilities_do_not_sum_to_one PASSED [  6%]
test/test_part1.py::TestFunctionEvaluatePMF::test_throws_when_passed_negatively_likely_outcomes PASSED [ 13%]
test/test_part1.py::TestFunctionEvaluatePMF::test_correct_for_two_dice_sum PASSED [ 20%]
test/test_part1.py::TestFunctionEvaluatePMF::test_correct_for_two_dice_diff PASSED [ 26%]
test/test_part1.py::TestFunctionEvaluatePMF::test_correct_for_rolling_doubles PASSED [ 33%]
test/test_part1.py::TestFunctionExpectedValue::test_throws_when_probabilities_do_not_sum_to_one PASSED [ 40%]
test/test_part1.py::TestFunctionExpectedValue::test_throws_when_passed_negatively_likely_outcomes PASSED [ 46%]
test/test_part1.py::TestFunctionExpectedValue::test_throws_when_passed_empty_pmf PASSED [ 53%]
test/test_part1.py::TestFunctionExpectedValue::test_correct_for_two_dice_sum PASSED [ 60%]
test/test_part1.py::TestFunctionExpectedValue::test_correct_for_two_dice_diff PASSED [ 66%]
test/test_part1.py::TestFunctionVariance::test_throws_when_probabilities_do_not_sum_to_one PASSED [ 73%]
test/test_part1.py::TestFunctionVariance::test_throws_when_passed_negatively_likely_outcomes PASSED [ 80%]
test/test_part1.py::TestFunctionVariance::test_throws_when_passed_empty_pmf PASSED [ 86%]
test/test_part1.py::TestFunctionVariance::test_correct_for_two_dice_sum PASSED [ 93%]
test/test_part1.py::TestFunctionVariance::test_correct_for_two_dice_diff PASSED [100%]

============================== 15 passed in 0.32s ==============================
```

QUESTION: Why does the PMF for the sum of two dice look the way it does? Can you explain why a value of 7 is more likely than a value of 10?

- The central values have more than one way to be rolled. A 2 or a 12 result only has one roll that can achieve them ((1,1) and (6,6)).
- There are 6 rolls that will result in a 7, while there are only 3 that will result in a 10. This explains why the probability of rolling a 7 (~ 0.16) is twice as much as rolling a 10 (~0.08)

Section 2



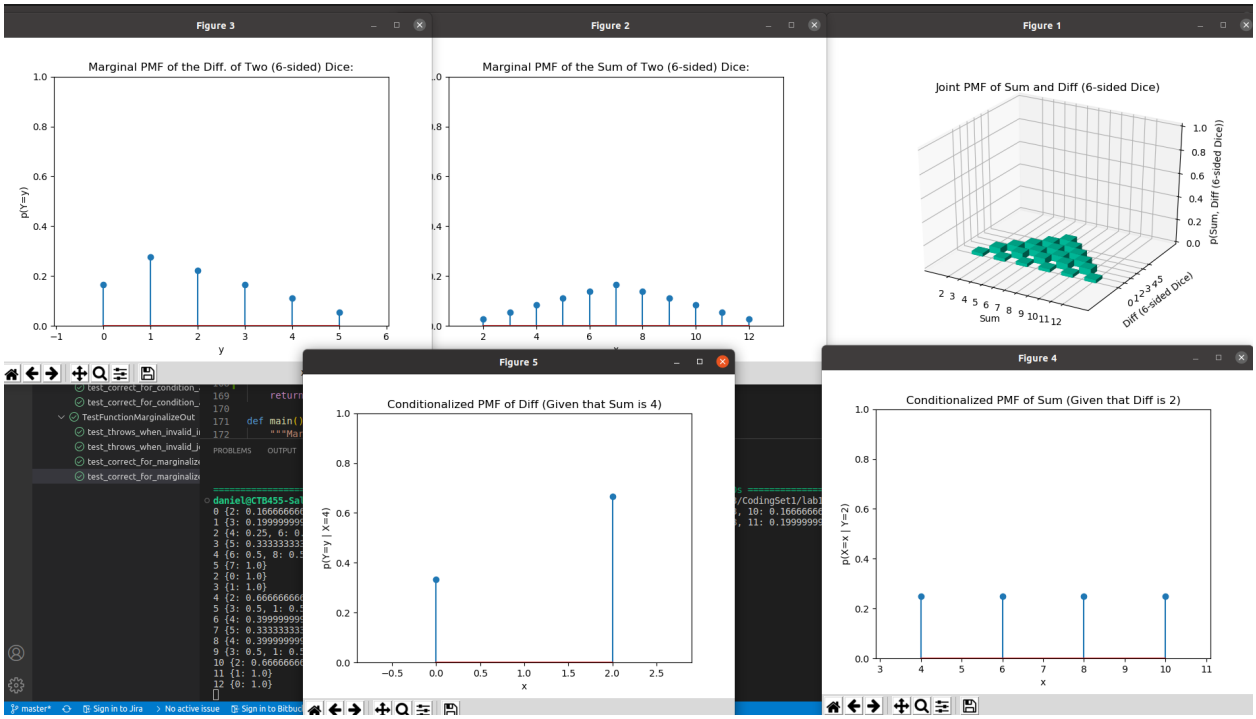

```
daniel@CTB455-Salmon:~/EC EN 633/CodingSet1/lab1-prob-review$ /bin/python "/home/daniel/EC EN 633/CodingSet1/lab1-prob-review/probability_review/part2.py"
Expected value (Joint of sum/diff) with 6-sided Dice: [7.         1.94444444]
Covariance (Joint of sum/diff) with 6-sided Dice: [[5.83333333 0.        ]
 [0.         2.05246914]]
Expected value (Joint of sum/sum_plus_one) with 6-sided Dice: [7. 8.]
Covariance (Joint of sum/sum_plus_one) with 6-sided Dice: [[5.83333333 5.83333333]
 [5.83333333 5.83333333]]
Expected value (Joint of sum/weightedsum) with 6-sided Dice: [ 7.   12.25]
Covariance (Joint of sum/weightedsum) with 6-sided Dice: [[ 5.83333333 10.20833333]
 [10.20833333 21.14583333]]
```

```
daniel@CTB455-Salmon:~/EC EN 633/CodingSet1/lab1-prob-review$ pytest -v -rN --tb=no --no-header test/test_part2.py
============================= test session starts ==============================
collected 17 items

test/test_part2.py::TestFunctionEvaluateJointPMF::test_throws_when_probabilities_do_not_sum_to_one PASSED [  5%]
test/test_part2.py::TestFunctionEvaluateJointPMF::test_throws_when_passed_negatively_likely_outcomes PASSED [ 11%]
test/test_part2.py::TestFunctionEvaluateJointPMF::test_correct_for_two_dice_sum_and_two_dice_difference PASSED [ 17%]
test/test_part2.py::TestFunctionEvaluateJointPMF::test_correct_for_two_dice_sum_and_two_dice_sum_plus_one PASSED [ 23%]
test/test_part2.py::TestFunctionEvaluateJointPMF::test_correct_for_two_dice_sum_and_weighted_two_dice_sum PASSED [ 29%]
test/test_part2.py::TestFunctionExpectedValue::test_throws_when_probabilities_do_not_sum_to_one PASSED [ 35%]
test/test_part2.py::TestFunctionExpectedValue::test_throws_when_passed_negatively_likely_outcomes PASSED [ 41%]
test/test_part2.py::TestFunctionExpectedValue::test_throws_when_passed_empty_pmf PASSED [ 47%]
test/test_part2.py::TestFunctionExpectedValue::test_correct_for_two_dice_sum_and_two_dice_difference PASSED [ 52%]
test/test_part2.py::TestFunctionExpectedValue::test_correct_for_two_dice_sum_and_two_dice_sum_plus_one PASSED [ 58%]
test/test_part2.py::TestFunctionExpectedValue::test_correct_for_two_dice_sum_and_weighted_two_dice_sum PASSED [ 64%]
test/test_part2.py::TestFunctionCovariance::test_throws_when_probabilities_do_not_sum_to_one PASSED [ 70%]
test/test_part2.py::TestFunctionCovariance::test_throws_when_passed_negatively_likely_outcomes PASSED [ 76%]
test/test_part2.py::TestFunctionCovariance::test_throws_when_passed_empty_pmf PASSED [ 82%]
test/test_part2.py::TestFunctionCovariance::test_correct_for_two_dice_sum_and_two_dice_difference PASSED [ 88%]
test/test_part2.py::TestFunctionCovariance::test_correct_for_two_dice_sum_and_two_dice_sum_plus_one PASSED [ 94%]
test/test_part2.py::TestFunctionCovariance::test_correct_for_two_dice_sum_and_weighted_two_dice_sum PASSED [100%]

============================== 17 passed in 0.38s ==============================
```

QUESTION: For each pair of jointly distributed random variables, are the two variables correlated with one another? How do you know? Does this make sense?

- 1st pair, not correlated because the inverse diagonals are 0 for the covariance matrix.
- 2nd, 3rd pair, they are correlated at least somewhat because they have non-zero values on the inverse diagonals
- This makes sense because the sum and difference won't be correlated in any linear way, while the sum/sum_plus_one and sum/weighted_sum are definitely correlated.
    - To expand, for the first pair bigger sum does not necessarily mean a bigger difference or vice versa. For example, a sum of 12 would result in a difference of 0, so would a sum of 2.

Section 3




```
daniel@CTB455-Salmon:~/EC EN 633/CodingSet1/lab1-prob-review$ /bin/python "/home/daniel/EC EN 633/CodingSet1/lab1-prob-review/probability_review/part3.py"
 0 {2: 0.16666666666666663, 4: 0.16666666666666663, 6: 0.16666666666666663, 8: 0.16666666666666663, 10: 0.16666666666666663, 12: 0.16666666666666663}
 1 {3: 0.19999999999999998, 5: 0.19999999999999998, 7: 0.19999999999999998, 9: 0.19999999999999998, 11: 0.19999999999999998}
 2 {4: 0.25, 6: 0.25, 8: 0.25, 10: 0.25}
 3 {5: 0.3333333333333333, 7: 0.3333333333333333, 9: 0.3333333333333333}
 4 {6: 0.5, 8: 0.5}
 5 {7: 1.0}
 2 {0: 1.0}
 3 {1: 1.0}
 4 {2: 0.6666666666666666, 0: 0.3333333333333333}
 5 {3: 0.5, 1: 0.5}
 6 {4: 0.39999999999999997, 2: 0.39999999999999997, 0: 0.19999999999999998}
 7 {5: 0.3333333333333333, 3: 0.3333333333333333, 1: 0.3333333333333333}
 8 {4: 0.39999999999999997, 2: 0.39999999999999997, 0: 0.19999999999999998}
 9 {3: 0.5, 1: 0.5}
 10 {2: 0.6666666666666666, 0: 0.3333333333333333}
 11 {1: 1.0}
 12 {0: 1.0}
```

```
daniel@CTB455-Salmon:~/EC EN 633/CodingSet1/lab1-prob-review$ pytest -v -rN --tb=no --no-header test/test_part3.py
============================= test session starts ==============================
collected 8 items

test/test_part3.py::TestFunctionMarginalizeOut::test_throws_when_invalid_indicator_passed_in PASSED [ 12%]
test/test_part3.py::TestFunctionMarginalizeOut::test_throws_when_invalid_joint_pmf_received PASSED [ 25%]
test/test_part3.py::TestFunctionMarginalizeOut::test_correct_for_marginalize_out_difference_from_sum_and_difference PASSED [ 37%]
test/test_part3.py::TestFunctionMarginalizeOut::test_correct_for_marginalize_out_sum_from_sum_and_difference PASSED [ 50%]
test/test_part3.py::TestFunctionConditionAgainst::test_throws_when_invalid_indicator_passed_in PASSED [ 62%]
test/test_part3.py::TestFunctionConditionAgainst::test_throws_when_invalid_joint_pmf_received PASSED [ 75%]
test/test_part3.py::TestFunctionConditionAgainst::test_correct_for_condition_against_difference_from_sum_and_difference PASSED [ 87%]
test/test_part3.py::TestFunctionConditionAgainst::test_correct_for_condition_against_sum_from_sum_and_difference PASSED [100%]

============================== 8 passed in 0.30s ===============================
```

QUESTION: Do the marginal PMFs match the PMFs you generated in part 1?

- They do!

QUESTION: Please describe the output of the conditional PMFs. Why does the output make sense?

- For the case of the Conditionalized PMF of Diff (Given that Sum is 4)

    - There are only 3 possibilities of rolls: (2,2), (3,1) and (1,3) that result in a sum of 4. The differences are 0 and 2 with respective probabilities of ⅓ and ⅔ which is reflected in the plot.
- For the case of the Conditionalized PMF of Sum (Given that Diff is 2)
    - There are the following possibilities with the given conditio: (2,4), (4,2), (6,4), (4,6), (1,3), (3,1), (5,3), (3,5). Which result in the following sums with equal probability: 4, 6, 8, 10. The plot reflects this.