UNIVERSITA DEGLI STUDI DI GENOVA

DIBRIS

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY, BIOENGINEERING AND SYSTEM ENGINEERING

Research Track 2

Third Assignment

Statistical Analysis

Author

Vahid Bagherian

6016232

May 2024

1) Introduction

In the final task of Research Track 2, a detailed statistical analysis is performed on the first assignment from Research Track 1. The first assignment of Research Track 1 was simulating and controlling a mobile robot in an environment. The environment contained some golden boxes and the robot was supposed to gather all the boxes next to each other in one place. In this report, the performance of my code is compared with the performance of one of my colleagues' code which does the same task. The evaluation of performance is based on the hypotheses which are described in the next section. The parameter that determines which algorithm performs better is the time it takes for the robot to gather all the golden boxes next to each other and also the number of successes and failures of each algorithm in detecting and gathering the golden boxes when the number of boxes in the environment changes.

2) Hypothesis

Hypothesis Testing is a statistical tool that is usually used in scientific research to reach a conclusion about the parameters of a data set (mean, variance, etc.).

The hypothesis for a given problem is described by two competing hypotheses: **Null hypothesis (H_0)** which states that the two methods that are being compared are equally good (based on the evaluation methods applied later). On the other side, **Alternative hypothesis (H_a)** is defined which states that one method is superior in comparison to the other method. If one of these hypotheses is true, the other one is false, and vice versa. One method to prove that the alternative hypothesis is true and one method is better than the other method is to reject the null hypothesis. Consequently, as a first step in analyzing the two codes in this report, the null and alternative hypotheses should be defined as follows:

- Null Hypothesis (H_0): The results obtained from my code and my colleague's code have the same performance and they are similar to each other. In other words, we can not determine which algorithm is faster than the other one in gathering the boxes and the average time for this task is equal in both algorithms ($\mu_A = \mu_B$).
- Alternative hypothesis (H_a): My colleague's algorithm is faster than my algorithm in gathering the boxes next to each other under different circumstances. Consequently, the average time it takes for my robot to gather all the boxes is more in comparison with my colleague's code ($\mu_A > \mu_B$).

In the first step to test the null hypothesis, sample test data must be gathered. The number of test data should be enough so that it gives proper, accurate, and trustworthy results. There are different methods to check and compare the performance of two algorithms, such as Z-test, T-test, two sampled T-test, paired T-test, etc. There are some basic preconditions for each method so that we can use that specific method for analysis. Z-test is usually used when the population and its mean and standard deviation are available. To use the T-test, at least one of the characteristics of the population are known (mean, standard deviation). In the case of this report, the population is not available and we are just comparing two different algorithms that complete a task. For this analysis, two sampled T-test and paired T-test can be useful.

3) Experiment and results

In order to obtain the test data for the analysis, the same environments are created for both algorithms. Each algorithm is tested in different environments with 3, 4, 5, 6, and 7 golden boxes to check how fast it gathers all the boxes and how accurate each algorithm is in detecting the boxes and gathering them. The algorithms are executed 6 times in each environment configuration which makes an overall 30 simulations for each algorithm. To create the exact same environment for each algorithm and validate the comparison, we did not use random generation of the boxes. Consequently, the positions of the boxes are the same in an environment with the same number of boxes. We have decided to apply a paired T-test on the extracted data. We could have also used a two sample T-test if we simulated both algorithms 30 times with the same number of randomly generated golden boxes in the environment.

This method should be implemented as follows:

1. Calculate the difference ($d_i = y_i - x_i$) between the two observations on each simulation pair of two algorithms, making sure to distinguish between positive and negative differences
2. Calculate the mean difference ($\bar{d}$).
3. Calculate the standard deviation of the differences, s_d, and use this to calculate the standard error of the mean difference ($SE(\bar{d}) = \frac{s_d}{\sqrt{n}}$)
4. Calculate the t-statistic, which is given by $T = {\bar{d}}/{SE(\bar{d})}$. Under the null hypothesis, this statistic follows a t distribution with n − 1 degrees of freedom.
5. Use tables of the t-distribution to compare your value for T to the t_{n-1} distribution. This will give the p-value for the paired t-test.

In Table 1, the results of my code and my colleague's code simulation time and success are presented under different circumstances. It can be seen that both codes were completely successful in detecting and gathering the boxes in all cases that were investigated. In some cases with a small number of gold boxes, my code misplaced one of the boxes at first but later during the simulation the same box was picked up again and placed next to other boxes. This case can not be considered a failure as the robot finally gathers all the boxes next to each other which is the only criterion for success and failure of the algorithm. However, as the robot picks up and moves a box twice, it finally affects the simulation time of the algorithm. Consequently, the effect of this little mistake is projected in the performance evaluation of the algorithms.

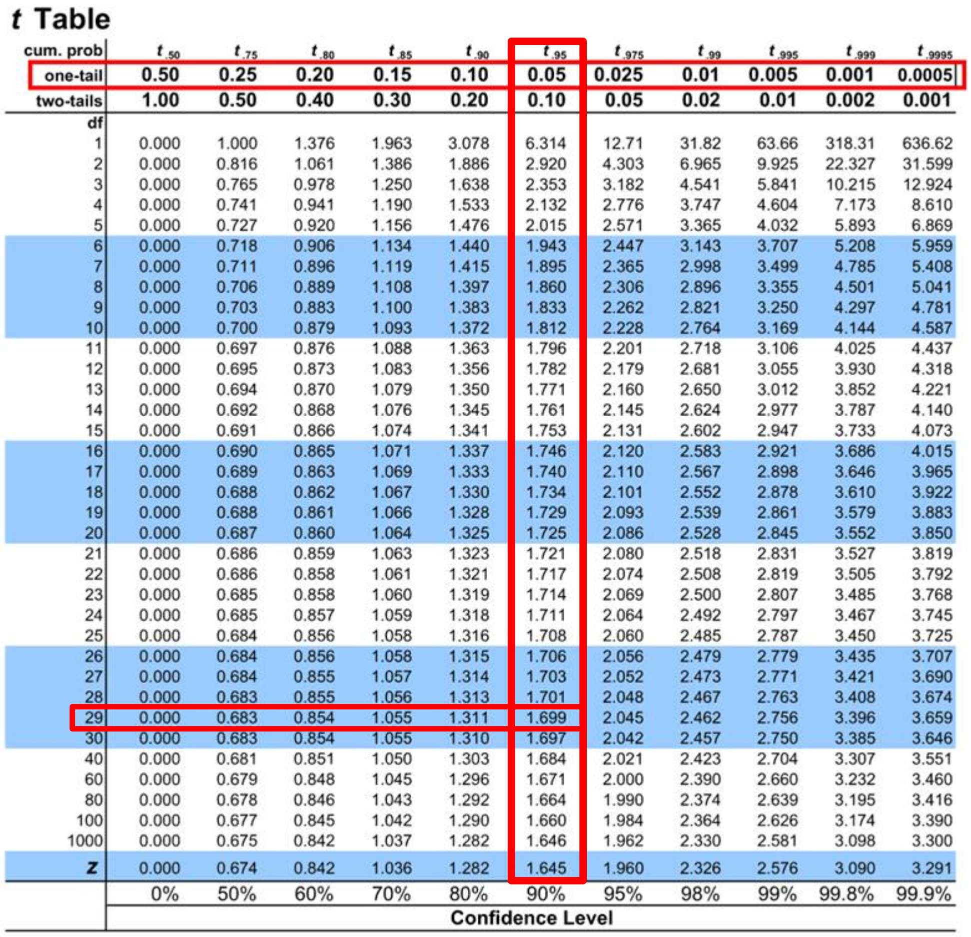
Table 1) Simulation time and successfulness of my code versus my colleague's code

Test Number	Number of boxes	my Code Tme	Colleague Code Time	Difference	success of my code	success of my colleague code
1	3	140.27	30.63	109.64	successful	successful
2	3	140.08	29.41	110.67	successful	successful
3	3	136.24	26.56	109.68	successful	successful
4	3	138.39	29.36	109.03	successful	successful
5	3	137.39	30.26	107.13	successful	successful
6	3	140.61	29.91	110.70	successful	successful
7	4	167.14	49.6	117.54	successful	successful
8	4	166.08	49.08	117.00	successful	successful
9	4	165.50	46.77	118.73	successful	successful
10	4	164.88	46.99	117.89	successful	successful
11	4	164.20	50.69	113.51	successful	successful
12	4	164.79	49.69	115.10	successful	successful
13	5	197.27	58.04	139.23	successful	successful
14	5	191.02	58.24	132.78	successful	successful
15	5	195.93	52.31	143.62	successful	successful
16	5	194.19	51.5	142.69	successful	successful
17	5	193.61	52.31	141.30	successful	successful
18	5	197.97	56.88	141.09	successful	successful
19	6	197.63	70.54	127.09	successful	successful
20	6	193.74	70.21	123.53	successful	successful
21	6	201.59	67.71	133.88	successful	successful
22	6	202.20	68.29	133.91	successful	successful
23	6	201.32	67.47	133.85	successful	successful
24	6	201.32	70.77	130.55	successful	successful
25	7	229.37	75.72	153.65	successful	successful
26	7	230.84	72.55	158.29	successful	successful
27	7	231.62	80.8	150.82	successful	successful
28	7	231.96	77.56	154.40	successful	successful
29	7	229.16	82.56	146.60	successful	successful
30	7	230.56	89.2	141.36	successful	successful
Mean		185.90	56.39	129.51		
Standard deviation		31.91	17.86	15.76		
SE		5.82609968	3.260354686	2.87668271		
t		31.9075059	17.29474411	45.0203664		

It should be noted that in order to see the simulation time for each algorithm, a code is added to both algorithms which calculates the simulation time. The main function is written as followed:

```
def main ():

    current_time = time.time()

    … # The rest of the code and algorithm execution

    elapsed_time = time.time() - current_time

    print("Elapsed time:", elapsed_time)
```

function Based on the observations and calculations in Table 1, it can be seen that the value of t for the paired T-test is equal to 45.02. A paired T-test with a confidence of 99.5% is applied to the data with a degree of freedom of 29. Based on one tail T-test table represented in Figure 1, it can be seen that our computed t-value should be compared with 1.699.



***t* Table**

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	0.50	0.25	0.20	0.15	0.10	0.05	0.025	0.01	0.005	0.001	0.0005
two-tails	1.00	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01	0.002	0.001
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	0.000	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	0.000	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	0.000	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	0.000	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	0.000	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	0.000	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	0.000	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	0.000	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	0.000	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	0.000	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	0.000	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	0.000	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	0.000	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	0.000	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	0.000	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	0.000	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	0.000	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	0.000	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.421	3.690
28	0.000	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	0.000	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.396	3.659
30	0.000	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.385	3.646
40	0.000	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.307	3.551
60	0.000	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	3.232	3.460
80	0.000	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	3.195	3.416
100	0.000	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	3.174	3.390
1000	0.000	0.675	0.842	1.037	1.282	1.646	1.962	2.330	2.581	3.098	3.300
z	0.000	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.090	3.291
	0%	50%	60%	70%	80%	90%	95%	98%	99%	99.8%	99.9%
	Confidence Level										

Figure 1) one tailed T-test table

If the t value we computed is larger than 1.699, then the H_0 hypothesis is rejected and H_a hypothesis is accepted. As can be seen, the t value we calculated is much larger than 1.699 which obviously rejects the null hypothesis and proves that my colleague's code is much faster than my code in completing this task. Based on Figure 2, it can be seen from the bar chart that in all cases the simulation time for my colleague's code was much smaller than my code's simulation time.

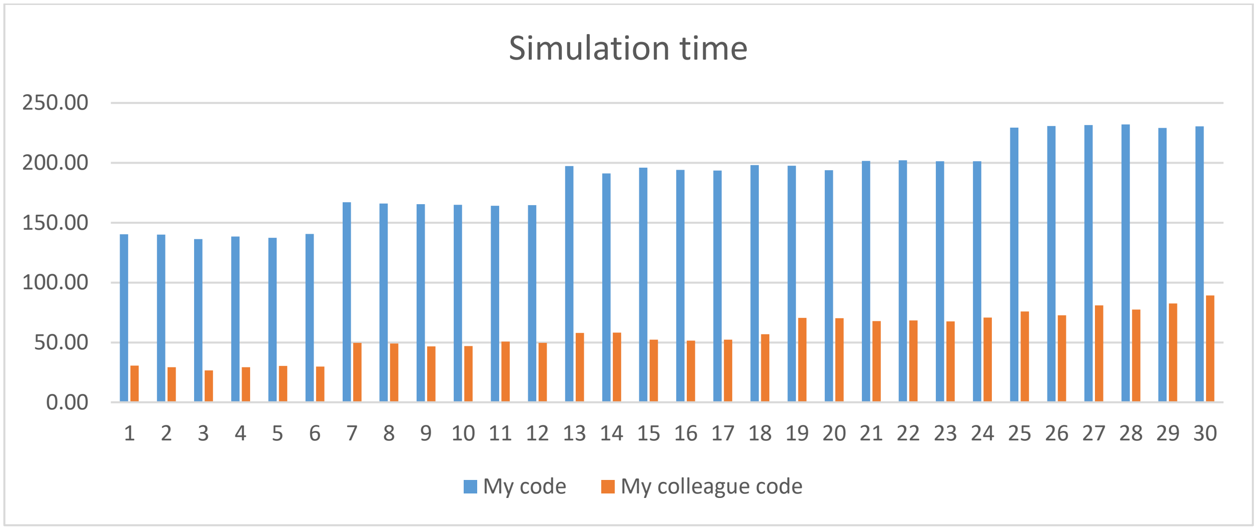


Figure 2) bar chart showing the simulation time comparison between the two algorithms