Aiden Bai

Kealey Sitler

AP Statistics

May 22, 2023

Research Question and Rationale

How do the performance outcomes of JavaScript frameworks categorized as "No Virtual DOM (Document Object Model)" and "Virtual DOM" based implementations compare when evaluated using the Mann Whitney U test in the context of the Keyed Results of the JS Framework Benchmark's Chrome 113 release?

To answer this question, it is imperative that we understand the context. JavaScript frameworks are used by web programmers to create websites, and effectively are the building block of the web. There are many options available, which has led to the creation of the JS Framework Benchmark, a community-driven, de facto standard for measuring and comparing various performance metrics, such as the time to load, create, update, and destroy parts of the user interface (UI) between JavaScript frameworks.

For this study, we will compare different framework designs. One framework design is the virtual DOM, which uses a "diffing algorithm" to make changes to a user interface. This design will help us segregate and create groups to compare the performance of virtual DOM to no virtual DOM. The results of this study will benefit JavaScript framework authors and web programmers to gain a better understanding of JavaScript framework designs.

Study Design

For the following study, we define *implementation* as the canonical usage of a JavaScript Framework. Implementations use *rendering engines* to display interfaces for websites. A popular rendering engine is the virtual DOM, which we will use for analysis in this study.

We will use a subset of the JS Framework Benchmark dataset. The benchmark contains several *benchmarks*, or tests to determine the performance of a certain implementation. Each cell in the table represents the sample mean of $n = 10$ samples of running the benchmark on the implementation.

A *slowdown* value, which is calculated by dividing the benchmark cell value with the fastest benchmark cell value in its row. For each slowdown value, a geometric mean is calculated for each implementation in order to determine the rank of each implementation. The geometric mean can also be used to determine an indicator of performance relative to the fastest implementation (see Figure 1). For example, an implementation with the geometric mean of 2 is two times slower than an implementation with the geometric mean of 1.

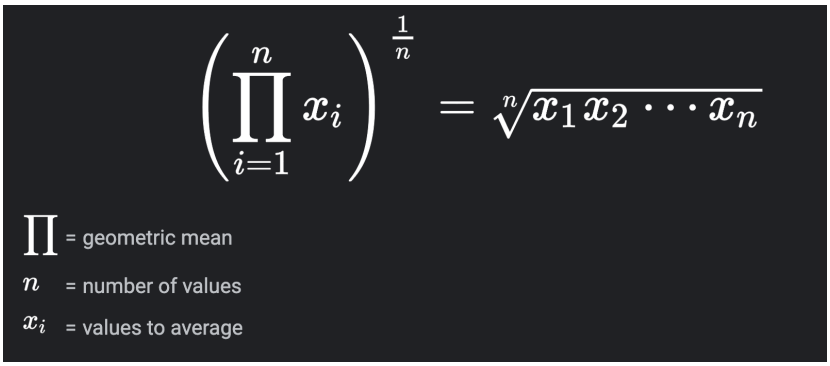

Figure 1. Where $n = 9$ (number of rows, which are benchmark tests), and x_n is a slowdown value for a particular row.

Furthermore, we exclude the vanillajs, vanillajs-1, vanillajs-wc, and any implementation that has the following issues explicitly marked in order to remove any externalities that do not conform to the research question's scope:

- #772 - Implementation uses manual DOM manipulations
- #796 - Implementation uses explicit requestAnimationFrame calls
- #800 - View state on the model
- #1139 - Implementation uses runtime code generation

Data Analysis

In Table 1, we see the raw data gathered for this study.

Group A	Geomean	Group B	Geomean
xania-v0.4.8	1.07	blockdom-v0.9.26	1.1
solid-v1.5.4	1.1	inferno-v7.4.8	1.1
maverick-v0.23.1	1.11	ivi-v3.0.0	1.1
malina-v0.7.0-alpha	1.11	million-v2.1.2	1.12
1more-v0.1.18	1.12	mimbl-v0.10.4	1.15
vuerx-jsx-v0.2.0	1.15	marionette-v5.0.0-alpha.2	1.2
mobx-jsx-v0.14.0	1.16	hyperapp-v2.0.22	1.2
solid-store-v1.5.4	1.17	vue-v3.2.47	1.26
fntags-v0.3.3	1.18	domvm-v3.4.12	1.26
michijs-v1.0.4	1.19	crank-v0.4.1	1.27
ef-js-v0.16.2	1.19	etch-v0.14.1	1.29
lit-v2.6.1	1.23	bobril-v20.4.1	1.31
lighterhtml-v2.5.0	1.24	karyon-v1.6.0	1.31
endorphin-v0.5.2	1.27	reflex-v0.8.4	1.37
art-v0.1.7	1.29	whatsup-v2.6.0	1.38
mahal-v1.4.3	1.32	preact-v10.13.1	1.42
svelte-v3.58.0	1.32	dark-v0.23.0	1.48
hydro-js-v1.5.13	1.41	mithril-v2.2.2	1.5
neverland-v3.3.2	1.43	gyron-v0.0.16	1.55
hullo-v0.8.2	1.44	jotai-v17.0.1 + 1.7.2	1.61

heresy-v0.26.1	1.49	react-hooks-v18.2.0	1.75
angular-nozone-v15.0.1	1.53	helix-v0.0.10	1.75
angular-v15.0.1	1.6	react-tagged-state-v18.2.0 + 1.23.2	1.77
lwc-v2.7.3	1.64	react-rxjs-v18.2.0 + 0.10.4	1.77
uhydro-v1.0.7	1.71	react-recoil-v18.2.0 + 0.7.7	1.78
oldskull-v2.0.0	1.86	react-mobX-v17.0.1 + 5.15.4	1.79
marko-v4.12.3	1.87	react-redux-hooks-v18.2.0 + 8.0.5	1.81
ractive-v1.3.6	1.97	fre-v2.5.5	1.82
glimmer-2-v2.0.0-beta.20	2.09	skruv-v0.1.0	1.87
ember-v4.10.0	2.14	react-signalis-v18.2.0 + 0.0.8	1.89
michijs-map-v1.0.4	2.24	react-diagon-v18.2.0 + 0.14.3	1.89
forgo-v2.2.3	4.16	react-v18.2.0	1.89
		react-redux-hooks-immutable-v18.2.0 + 8.0.5	1.9
		rax-v0.6.7	1.94
		react-tracked-v18.2.0 + 1.7.11	1.95
		react-zustand-v18.2.0 + 4.3.6	1.95
		rescript-react-v0.10.3	1.97
		anansi-v0.14.0	1.99
		react-hooks-use-transition-v18.2.0	2.05
		valtio-v18.2.0 + 1.10.3	2.1
		reagent-v0.10	2.1
		react-redux-rematch-v18.2.0 + 8.0.5 + 2.2.0	2.16
		react-redux-v18.2.0 + 8.0.5	2.16
		apprun-v2.28.3	2.26
		react-focal-v18.2.0 + 0.9.0	2.31
		arrowjs-v1.0.0-alpha.9	2.67
		react-starbeam-v18.2.0 + 0.6.0	2.86
		choo-v6.13.0	4.66

Table 1. Raw data with labeled geometric means for this study

In order to process this data, we need to rank each row relative to the total. So we reformat this data into a single table. Then, we can calculate rank sum value (Table 2) by using the =RANK.AVG(CELL,DATA_COL,1) function in Google Sheets. The “RANK.AVG” function returns the rank of a number in a list of numbers: its size relative to other values in the list; if more than one value has the same rank, the average rank is returned.

Group	Geomean	Rank
A	1.07	1
B	1.1	3.5
B	1.1	3.5
B	1.1	3.5
A	1.1	3.5
A	1.11	6.5
A	1.11	6.5
B	1.12	8.5
A	1.12	8.5
B	1.15	10.5
A	1.15	10.5
A	1.16	12
A	1.17	13
A	1.18	14
A	1.19	15.5
A	1.19	15.5
B	1.2	17.5
B	1.2	17.5
A	1.23	19
A	1.24	20
B	1.26	21.5
B	1.26	21.5
A	1.27	23.5
B	1.27	23.5
A	1.29	25.5
B	1.29	25.5

B	1.31	27.5
B	1.31	27.5
A	1.32	29.5
A	1.32	29.5
B	1.37	31
B	1.38	32
A	1.41	33
B	1.42	34
A	1.43	35
A	1.44	36
B	1.48	37
A	1.49	38
B	1.5	39
A	1.53	40
B	1.55	41
A	1.6	42
B	1.61	43
A	1.64	44
A	1.71	45
B	1.75	46.5
B	1.75	46.5
B	1.77	48.5
B	1.77	48.5
B	1.78	50
B	1.79	51
B	1.81	52
B	1.82	53
A	1.86	54
A	1.87	55.5
B	1.87	55.5
B	1.89	58
B	1.89	58
B	1.89	58
B	1.9	60
B	1.94	61

B	1.95	62.5
B	1.95	62.5
B	1.97	64.5
A	1.97	64.5
B	1.99	66
B	2.05	67
A	2.09	68
B	2.1	69.5
B	2.1	69.5
A	2.14	71
B	2.16	72.5
B	2.16	72.5
A	2.24	74
B	2.26	75
B	2.31	76
B	2.67	77
B	2.86	78
A	4.16	79
B	4.66	80

Table 2: Raw data of Group A and B values, along with their rank sum values.

When graphing the distributions of the data (Figure 2), we can see that both are left skewed and have no obvious outliers. Since it is skewed, we need to use a resistant measure of center. In this case, the medians for Group A and B are 1.32 and 1.78 respectively, and the range is 3.09 and 3.56 respectively.

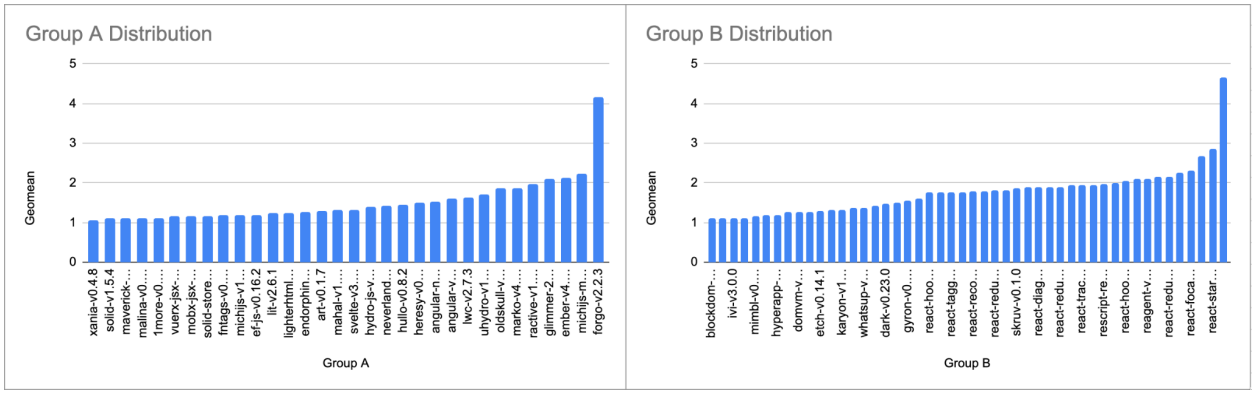

Figure 2. Group A and B distributions

Inference

Now that we have sourced our data, we proceed with the Mann Whitney U test (refer to Figure 3 for our "state" section). We first divide the implementations into two Groups: A, which holds groups that do not use a virtual DOM, and B, which uses a virtual DOM. Note that the original study design used the Wilcoxon Rank-Sum Test, but was scrapped because our sample sizes between groups were not equal. Our null hypothesis is that the test statistic for A will equal B, and our alternative hypothesis is that it will not equal. This test determines a difference in distributions based on the Mann Whitney U ranking.

Test Setup ("State")
$H_0: A = B$ $H_a: A \neq B$ A = Does not use virtual DOM B = Does use virtual DOM $\alpha = 0.05$ ($z = 1.96$)

Figure 3. State null hypothesis, alternative hypothesis, and A/B definitions

We then check whether the conditions (refer to Figure 4) fit our test. The test passes condition 1, as the dependent variable is measured on a continuous scale. The test passes condition 2, as both groups are categorical and independent by definition of the study and JS Framework Benchmark dataset. The test passes condition 3, as the observations are independent by definition of the dataset. Furthermore, the observations are not normally distributed, but follow the same shape (see Figure 2).

Mann Whitney U Test Conditions ("Plan")
1. The dependent variable should be measured on an ordinal scale or a continuous scale. 2. The independent variable should be two independent, categorical groups. 3. Observations should be independent. In other words, there should be no relationship between the two groups or within each group. 4. Observations are not normally distributed. However, they should follow the same shape (i.e. both are bell-shaped and skewed left).

Figure 4. Conditions for the Mann Whitney U Test

Once we have validated that our data passes the aforementioned conditions, we can proceed with our "Do" step (see Table 3). First, we calculate the Rank Sum for both groups, using conditional function in order to sum the values for each respective group. We then get the count of values for each group. After, we can calculate the U test statistic with $R - \frac{n_1(n_1+1)}{2}$, where R is the rank sum and n is the sample size. Since our sample size is greater than 30, we must assume normality. In order to find the P-value, we calculate the standard deviation of U to calculate the Z value with $z = \frac{x-\mu}{\sigma}$.

Calculating U test statistic and Z P-value ("Do")

Group	Rank Sum	n	U	Z
A	1032.5	32	504.5	-2.587814401
B	2207.5	48	1031.5	2.587814401
	=SUMIF(DATA_COL,GROUP,C2:C81)	=COUNTIF(DATA_COL,GROUP)	=G3-(H3*(H3+1))/2 $R - \frac{n_1(n_1+1)}{2}$	=(1/I6)*(I3-((H2*H3)/2)) $z = \frac{x-\mu}{\sigma}$
			SD of U	
			101.8	
			=SQRT((1/12)*(n_A*n_B)*(n_A+n_B+1))	

Table 3. Deriving the U test statistic and Z P-value from the rank sum and sample size. "FUNCTIONIF" functions are used in order to conditionally extract the correct data values from a combined column. The standard deviation of U ("SD of U") is calculated as the sample size exceeds the requirement for normality.

Upon executing the Mann Whitney U test, the test statistic (z) was calculated to be -2.59 for Group A (No Virtual DOM) and 2.59 for Group B (Virtual DOM). Given that the absolute value of the Z-score was greater than the critical value at the $\alpha = 0.05$ significance level ($z = 2.59 > 1.96$), we reject the null hypothesis that there is no difference between the distributions of the two groups. This indicates that there is a statistically significant difference between the performance outcomes of JavaScript frameworks with and without virtual DOM.

Obstacles

Initially, I approached this study with the Wilcoxon Rank-Sum Test, a similar test to the Mann Whitney U test. However, I discovered when conducting the Wilcoxon Rank-Sum Test that it requires equal sample sizes and variance between groups A and B. However, my data did

not conform to this requirement, as A had a much larger sample size than B. Therefore, I switched to the Mann Whitney U test, which doesn't require equal sample sizes between groups.

Another challenge I faced was determining the P-value of the test. Initially, I followed a reference study that assumed that sample sizes were less than 30. In their case, they used a table of critical values to compare against their derived test statistic. However, my sample size was greater than 30, so I needed to assume normality of the distribution according to the Central Limit Theorem. So, I calculated the standard deviation of the U to find the *z* test statistic, and I used that in order to figure out whether I needed to reject the null hypothesis or not.

Conclusion

In this study, we utilized the Mann Whitney U test to analyze performance differences between "No Virtual DOM" and "Virtual DOM" JavaScript frameworks. The findings showed a substantial influence of the choice of whether to use a virtual DOM on the efficiency of the frameworks. This observation is crucial for developers seeking the best performance in web development.

The outcomes of this research, however, do not conclude the discussion but set the stage for more detailed exploration. The distinct performance characteristics observed between the two framework categories suggest further investigation into the contributing factors. This could unlock potential strategies to enhance the performance of both categories.

Moreover, subsequent studies could target optimizing these factors, aspiring to boost the effectiveness of JavaScript frameworks. This could pave the way for the creation of more sophisticated frameworks, enhancing web interface performance and user experience.

This research direction can significantly benefit the web development sphere. By facilitating the evolution of more adept JavaScript frameworks, user interaction with the digital

world can be revolutionized. Besides user gratification, this also holds substantial business prospects in the increasingly digital economy.

Ultimately, this study reveals promising pathways for research into JavaScript framework performance. The findings emphasize the necessity of strategic decisions and consistent innovation to achieve superior web performance and user experience.

References

krausest. (n.d.). *GitHub - krausest/js-framework-benchmark: A comparison of the performance of a few popular javascript frameworks*. GitHub. Retrieved May 25, 2023, from https://github.com/krausest/js-framework-benchmark

The Mann-Whitney U-test -- Analysis of 2-Between-Group Data with a Quantitative Response Variable . (n.d.). University of Nebraska, Lincoln. https://psych.unl.edu/psycrs/handcomp/hcmann.PDF