



**Stan's  
Technologies**

# **Scientific Benchmark Report**

BoolMin2D vs SymPy Boolean Minimization

---

Experiment Date: 2026-01-07

Random Seed: 42

Total Test Cases: 3030

Statistical Significance Level:  $\alpha = 0.05$

*A Rigorous Statistical Analysis with Reproducibility Controls*

# EXPERIMENTAL SETUP

## SYSTEM CONFIGURATION

---

Python Version: 3.12.10  
Platform: Windows-11-10.0.26200-SP0  
Processor: Intel64 Family 6 Model 142 Stepping 12, GenuineIntel

## LIBRARY VERSIONS

---

SymPy: 1.14.0  
NumPy: 2.3.4  
SciPy: 1.16.3

## EXPERIMENTAL PARAMETERS

---

Random Seed: 42  
Tests per Configuration: 500  
Timing Warm-up Runs: 2  
Timing Repetitions: 5  
Significance Level ( $\alpha$ ): 0.05

## TEST DISTRIBUTIONS

---

- Sparse: 20% ones, 5% don't-cares (realistic digital logic)
- Dense: 70% ones, 5% don't-cares (stress test)
- Balanced: 50% ones, 10% don't-cares (neutral case)
- Minimal DC: 45% ones, 2% don't-cares (typical circuits)
- Heavy DC: 30% ones, 30% don't-cares (optimization test)
- Edge Cases: All-zeros, all-ones, checkerboard, single-minterm

## METHODOLOGY

---

1. Random K-maps generated with controlled distributions
2. Each algorithm executed with 2 warm-up runs
3. Best of 5 timed repetitions recorded
4. Logical equivalence verified using SymPy
5. Statistical significance tested using paired t-tests
6. Non-parametric Wilcoxon tests used as robustness check
7. Effect sizes computed using Cohen's d

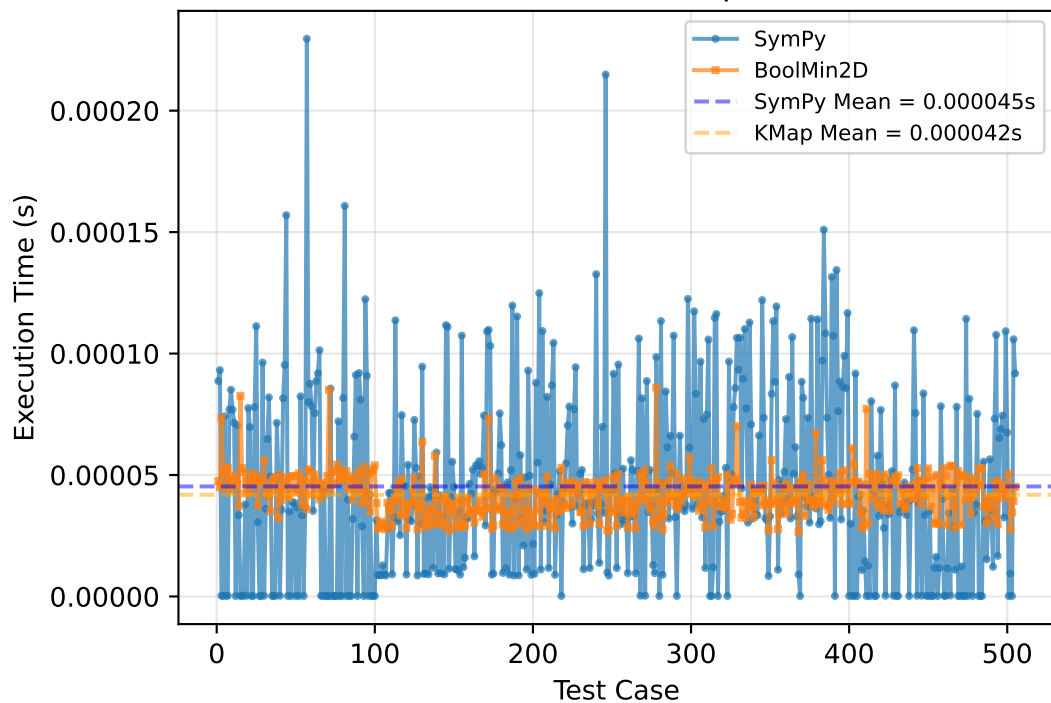
## REPRODUCIBILITY

---

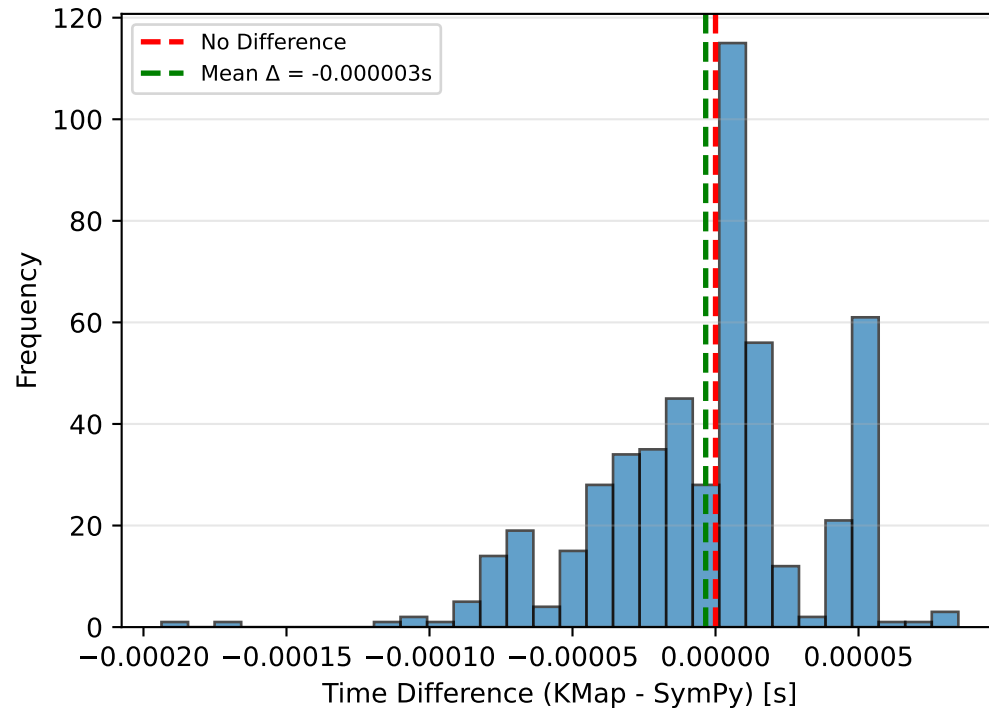
- To reproduce this experiment:
1. Set random seed: `random.seed(42)`
  2. Run with identical system configuration
  3. Use same library versions as documented above

# 2-Variable K-Map (POS Form)

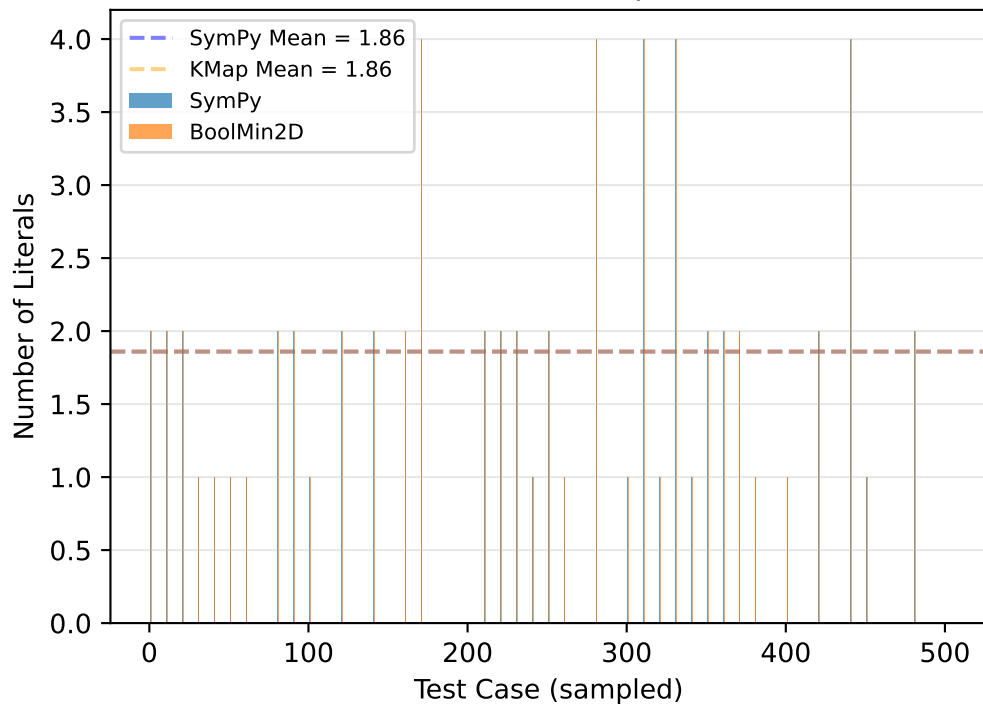
## Execution Time Comparison



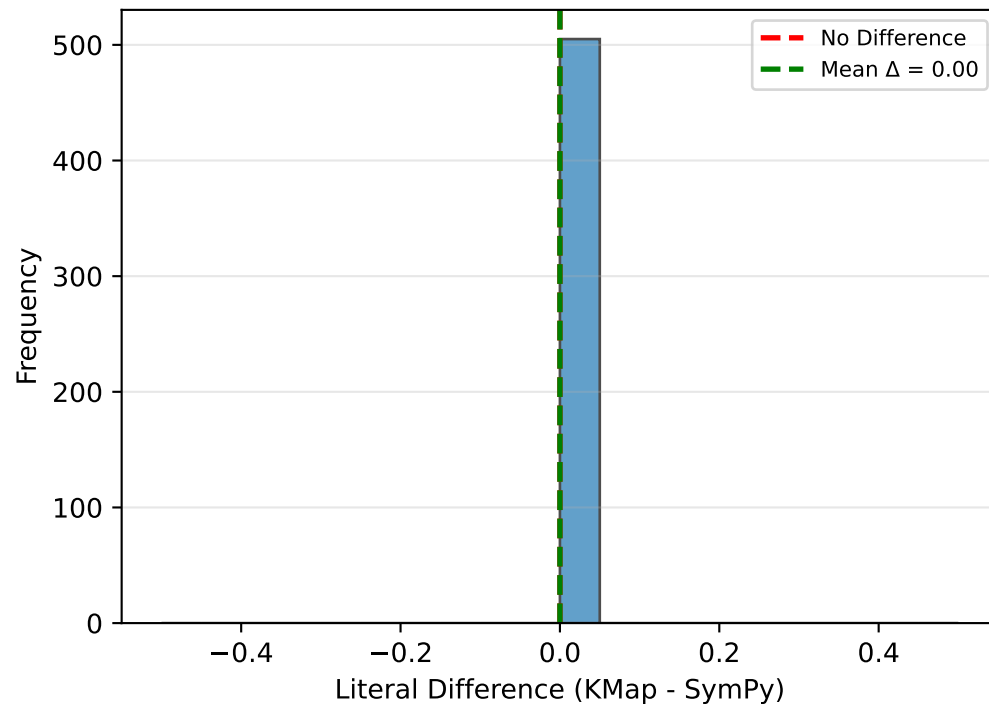
## Distribution of Time Differences



## Literal Count Comparison



## Distribution of Literal Differences



# STATISTICAL ANALYSIS: 2-Variable POS

## STATISTICAL INFERENCE REPORT

### ☐☐ CONSTANT FUNCTIONS DETECTED: 156/505 (30.9%)

These are unsimplifiable functions (e.g., all-zeros, all-ones) that both algorithms correctly identified. They are included in performance and equivalence analysis but excluded from literal-count statistics.

### 1. EXECUTION TIME ANALYSIS

Mean SymPy Time: 0.000045 s  
Mean BoolMin2D Time: 0.000042 s  
Mean Difference: -0.000003 s  
Std. Dev. ( $\Delta$ ): 0.000038 s  
95% CI: [-0.000007, -0.000000]

Paired t-test:  $t = -2.0413$ ,  $p = 0.041742$   
Wilcoxon test:  $W = 60261.0$ ,  $p = 0.269671$   
Effect Size (d): -0.0908 (negligible)

✓ **SIGNIFICANT: Time difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D is significantly faster than SymPy

### 2. SIMPLIFICATION QUALITY ANALYSIS

Analysis based on 349 non-constant functions:  
(156 constant function(s) excluded from this analysis)

Mean SymPy Literals: 1.86  
Mean KMap Literals: 1.86  
Mean Difference: +0.00  
Std. Dev. ( $\Delta$ ): 0.00  
95% CI: [nan, nan]

Paired t-test:  $t = \text{nan}$ ,  $p = \text{nan}$   
Wilcoxon test:  $W = 30537.5$ ,  $p = 1.000000$   
Effect Size (d): 0.0000 (negligible)

✗ **NOT SIGNIFICANT: No significant difference in simplification ( $p \geq 0.05$ )**  
→ Both algorithms achieve comparable minimization

### 3. OVERALL SCIENTIFIC CONCLUSION

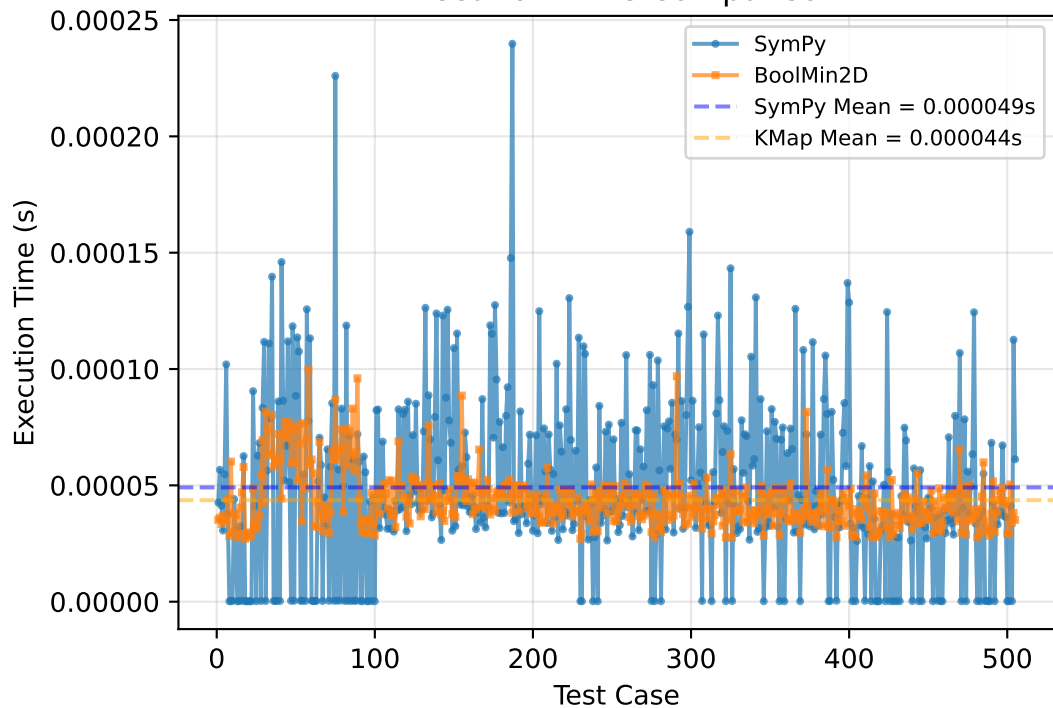
Performance difference is significant, but simplification is comparable.

Effect sizes: Time (negligible), Literals (negligible)

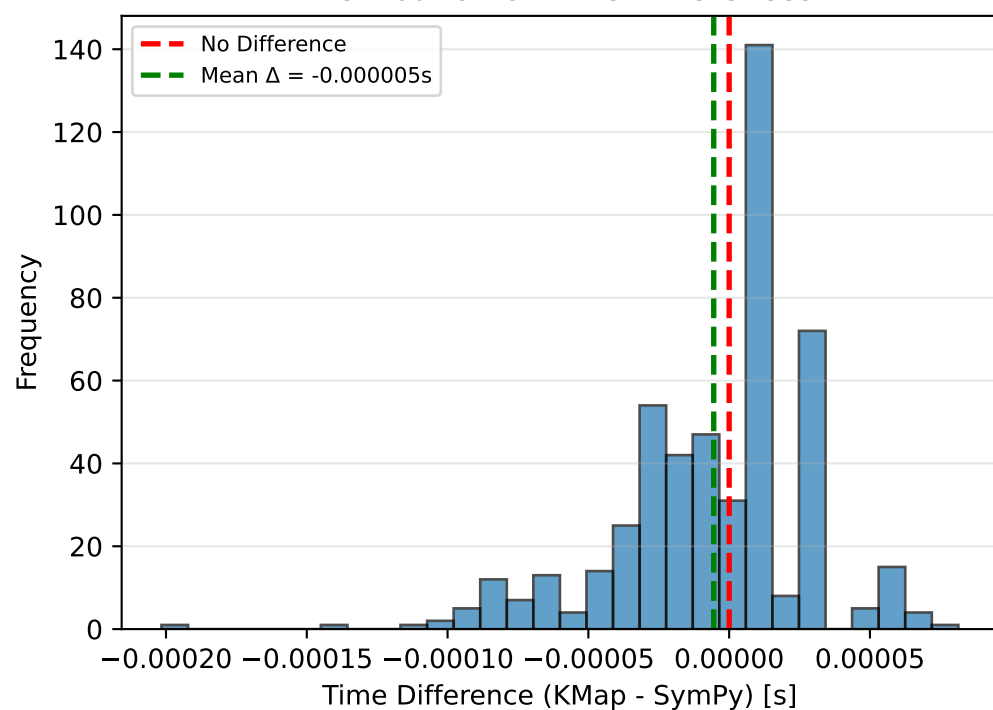
Note: 156 constant function(s) correctly handled by both algorithms.

# 2-Variable K-Map (SOP Form)

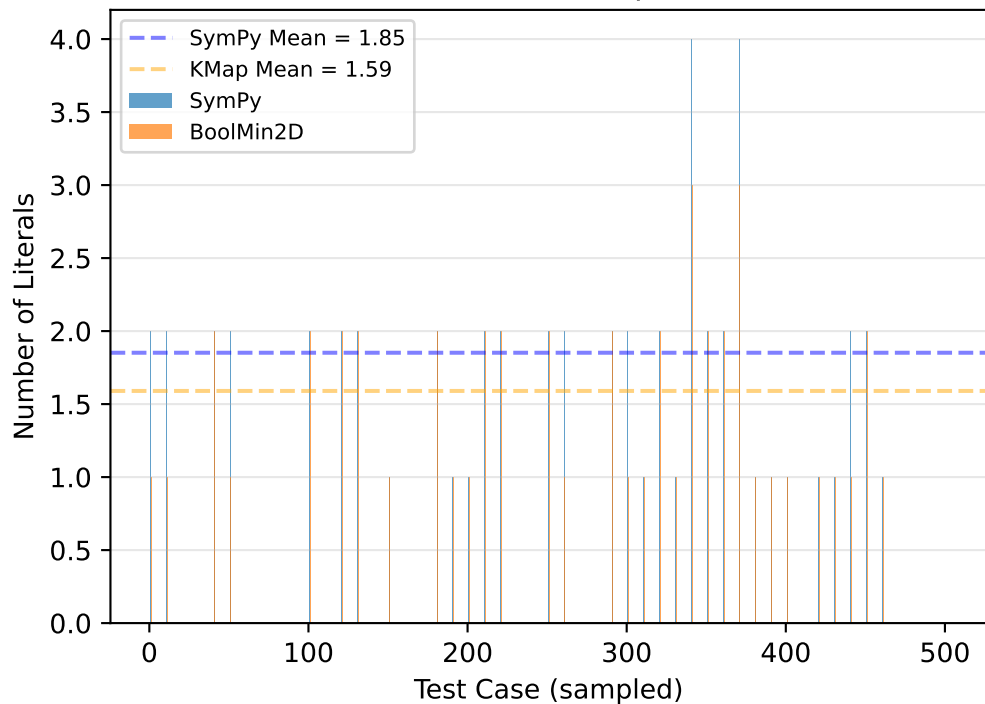
## Execution Time Comparison



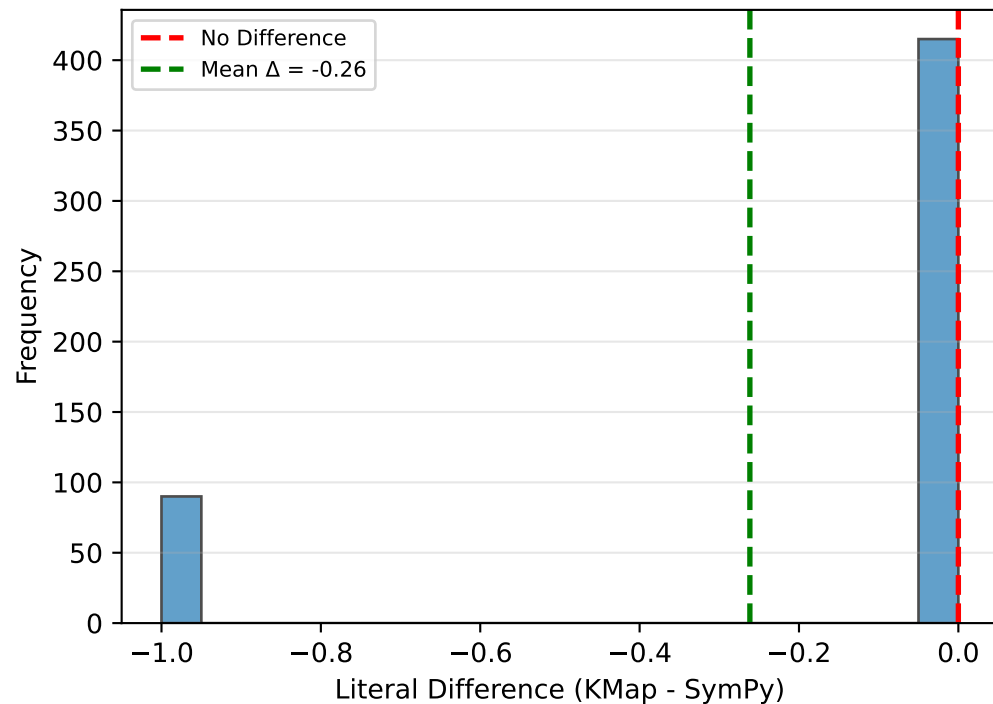
## Distribution of Time Differences



## Literal Count Comparison



## Distribution of Literal Differences



# STATISTICAL ANALYSIS: 2-Variable SOP

## STATISTICAL INFERENCE REPORT

### ☐☐ CONSTANT FUNCTIONS DETECTED: 161/505 (31.9%)

These are unsimplifiable functions (e.g., all-zeros, all-ones) that both algorithms correctly identified. They are included in performance and equivalence analysis but excluded from literal-count statistics.

### 1. EXECUTION TIME ANALYSIS

Mean SymPy Time: 0.000049 s  
Mean BoolMin2D Time: 0.000044 s  
Mean Difference: -0.000005 s  
Std. Dev. ( $\Delta$ ): 0.000035 s  
95% CI: [-0.000009, -0.000002]

Paired t-test:  $t = -3.5646$ ,  $p = 0.000399$   
Wilcoxon test:  $W = 56325.0$ ,  $p = 0.021251$   
Effect Size (d): -0.1586 (negligible)

✓ **SIGNIFICANT: Time difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D is significantly faster than SymPy

### 2. SIMPLIFICATION QUALITY ANALYSIS

Analysis based on 344 non-constant functions:  
(161 constant function(s) excluded from this analysis)

Mean SymPy Literals: 1.85  
Mean KMap Literals: 1.59  
Mean Difference: -0.26  
Std. Dev. ( $\Delta$ ): 0.44  
95% CI: [-0.31, -0.21]

Paired t-test:  $t = -11.0243$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 16192.5$ ,  $p = 0.000000$   
Effect Size (d): -0.5944 (medium)

✓ **SIGNIFICANT: Literal count difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D produces more minimal expressions

### 3. OVERALL SCIENTIFIC CONCLUSION

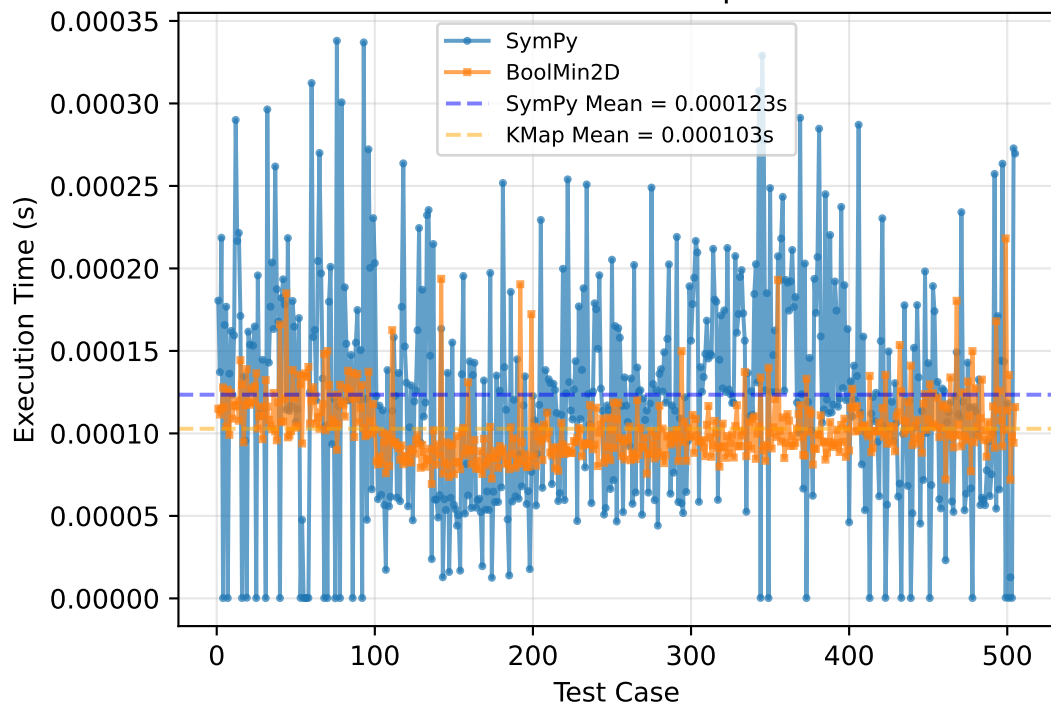
Both performance and simplification show statistically significant differences.

Effect sizes: Time (negligible), Literals (medium)

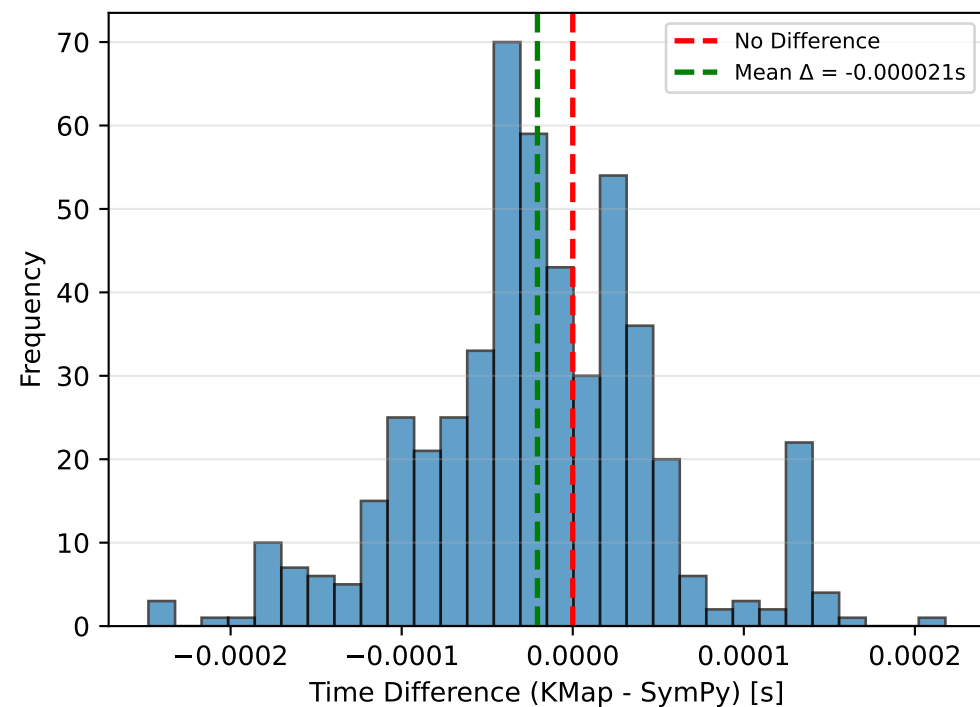
Note: 161 constant function(s) correctly handled by both algorithms.

# 3-Variable K-Map (POS Form)

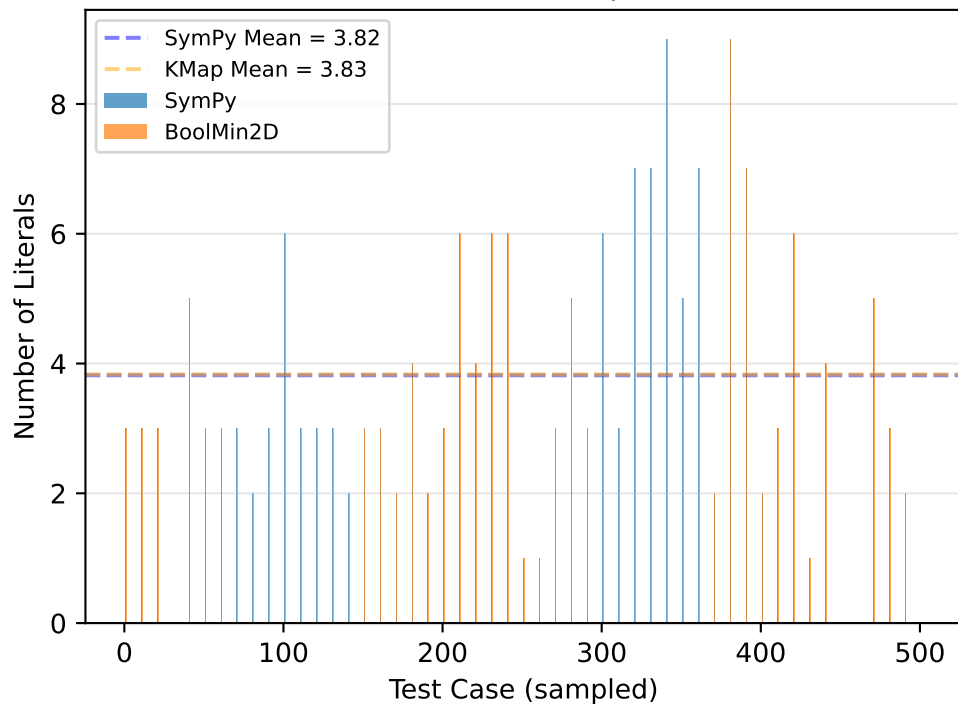
## Execution Time Comparison



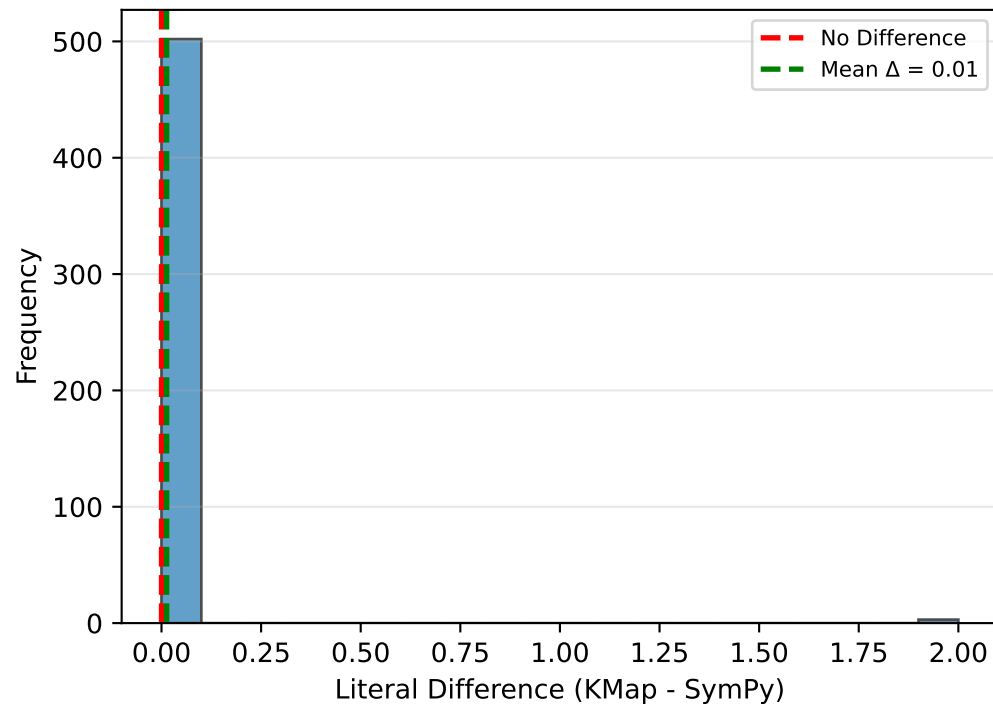
## Distribution of Time Differences



## Literal Count Comparison



## Distribution of Literal Differences



# STATISTICAL ANALYSIS: 3-Variable POS

## STATISTICAL INFERENCE REPORT

### ☐☐ CONSTANT FUNCTIONS DETECTED: 41/505 (8.1%)

These are unsimplifiable functions (e.g., all-zeros, all-ones) that both algorithms correctly identified. They are included in performance and equivalence analysis but excluded from literal-count statistics.

## 1. EXECUTION TIME ANALYSIS

Mean SymPy Time: 0.000123 s  
Mean BoolMin2D Time: 0.000103 s  
Mean Difference: -0.000021 s  
Std. Dev. ( $\Delta$ ): 0.000071 s  
95% CI: [-0.000027, -0.000014]

Paired t-test:  $t = -6.5619$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 41513.5$ ,  $p = 0.000000$   
Effect Size (d): -0.2920 (small)

✓ **SIGNIFICANT: Time difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D is significantly faster than SymPy

## 2. SIMPLIFICATION QUALITY ANALYSIS

Analysis based on 464 non-constant functions:  
(41 constant function(s) excluded from this analysis)

Mean SymPy Literals: 3.82  
Mean KMap Literals: 3.83  
Mean Difference: +0.01  
Std. Dev. ( $\Delta$ ): 0.16  
95% CI: [-0.00, 0.03]

Paired t-test:  $t = 1.7358$ ,  $p = 0.083264$   
Wilcoxon test:  $W = 53245.5$ ,  $p = 0.782193$   
Effect Size (d): 0.0806 (negligible)

✗ **NOT SIGNIFICANT: No significant difference in simplification ( $p \geq 0.05$ )**  
→ Both algorithms achieve comparable minimization

## 3. OVERALL SCIENTIFIC CONCLUSION

Performance difference is significant, but simplification is comparable.

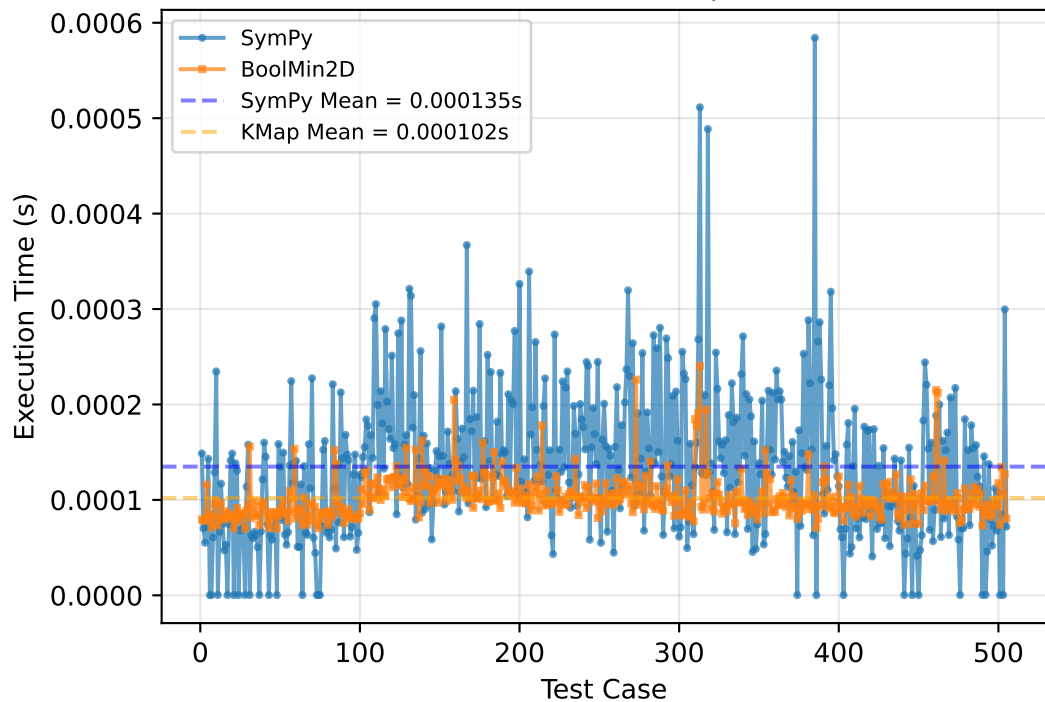
Effect sizes: Time (small), Literals (negligible)

Note: 41 constant function(s) correctly handled by both algorithms.

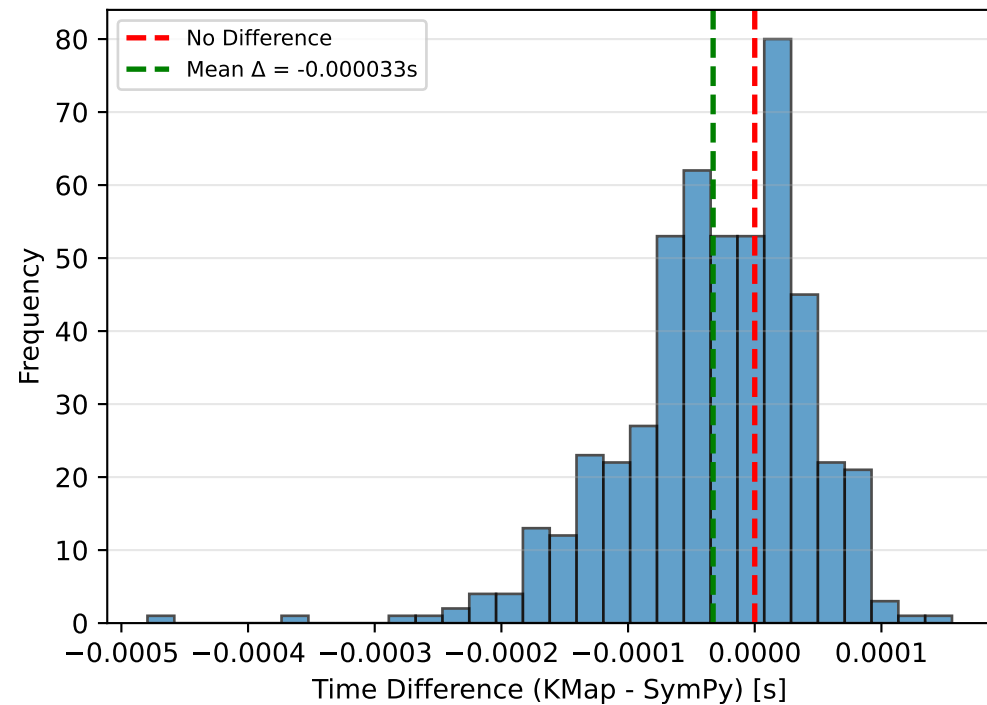


# 3-Variable K-Map (SOP Form)

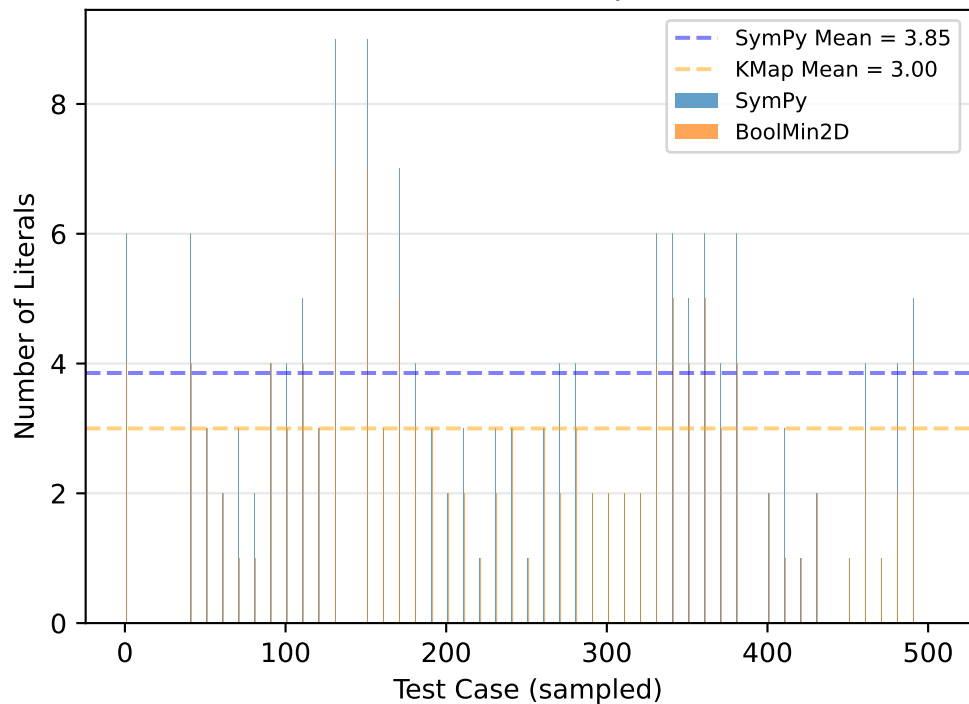
## Execution Time Comparison



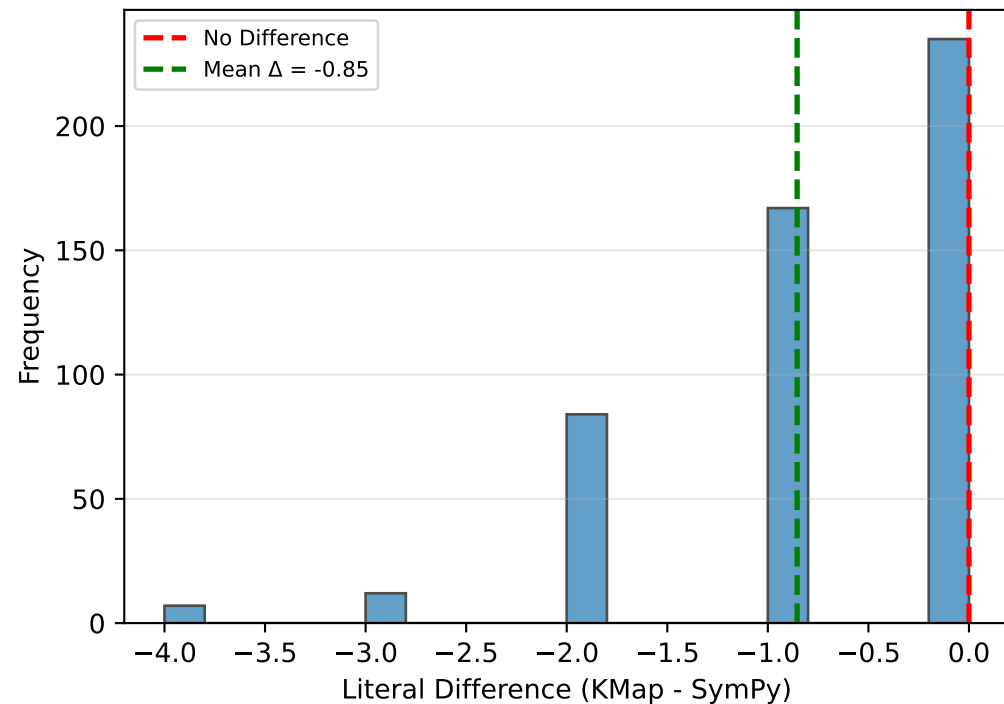
## Distribution of Time Differences



## Literal Count Comparison



## Distribution of Literal Differences



# STATISTICAL ANALYSIS: 3-Variable SOP

## STATISTICAL INFERENCE REPORT

### ☐☐ CONSTANT FUNCTIONS DETECTED: 38/505 (7.5%)

These are unsimplifiable functions (e.g., all-zeros, all-ones) that both algorithms correctly identified. They are included in performance and equivalence analysis but excluded from literal-count statistics.

## 1. EXECUTION TIME ANALYSIS

Mean SymPy Time: 0.000135 s  
Mean BoolMin2D Time: 0.000102 s  
Mean Difference: -0.000033 s  
Std. Dev. ( $\Delta$ ): 0.000074 s  
95% CI: [-0.000039, -0.000026]

Paired t-test:  $t = -10.0414$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 35193.0$ ,  $p = 0.000000$   
Effect Size (d): -0.4468 (small)

✓ **SIGNIFICANT: Time difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D is significantly faster than SymPy

## 2. SIMPLIFICATION QUALITY ANALYSIS

Analysis based on 467 non-constant functions:  
(38 constant function(s) excluded from this analysis)

Mean SymPy Literals: 3.85  
Mean KMap Literals: 3.00  
Mean Difference: -0.85  
Std. Dev. ( $\Delta$ ): 0.91  
95% CI: [-0.94, -0.77]

Paired t-test:  $t = -20.3901$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 9751.5$ ,  $p = 0.000000$   
Effect Size (d): -0.9435 (large)

✓ **SIGNIFICANT: Literal count difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D produces more minimal expressions

## 3. OVERALL SCIENTIFIC CONCLUSION

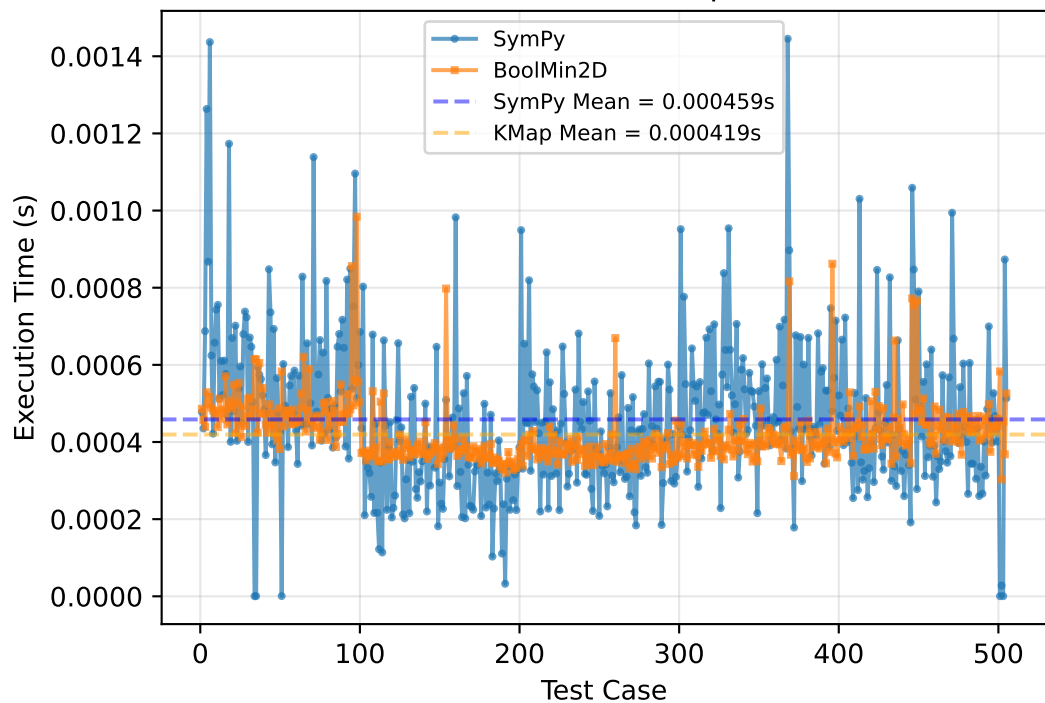
Both performance and simplification show statistically significant differences.

Effect sizes: Time (small), Literals (large)

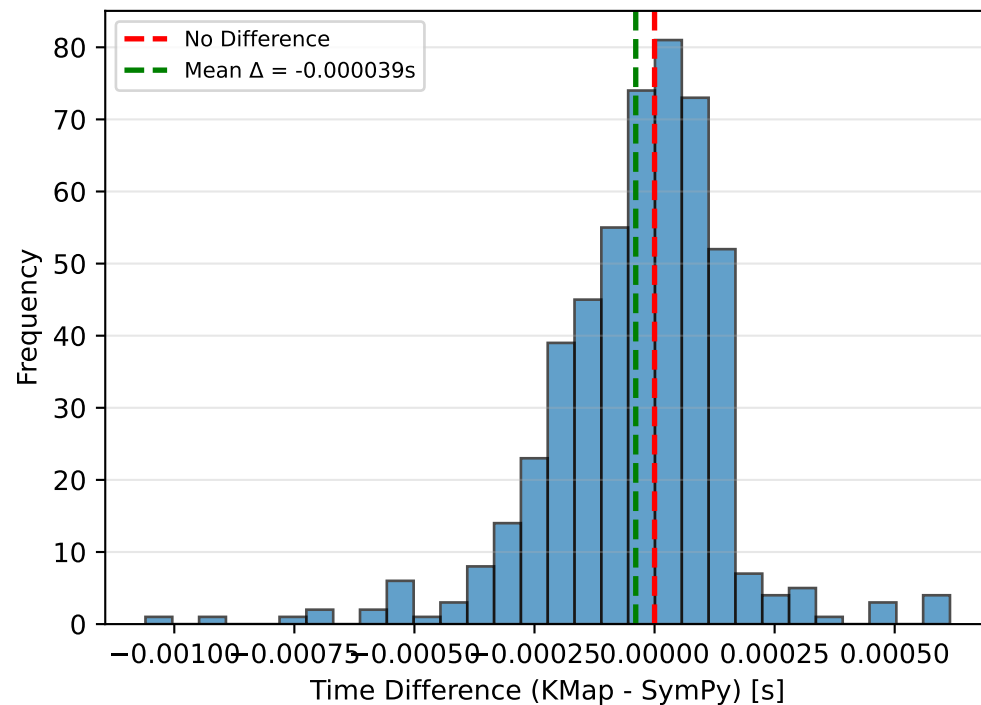
Note: 38 constant function(s) correctly handled by both algorithms.

# 4-Variable K-Map (POS Form)

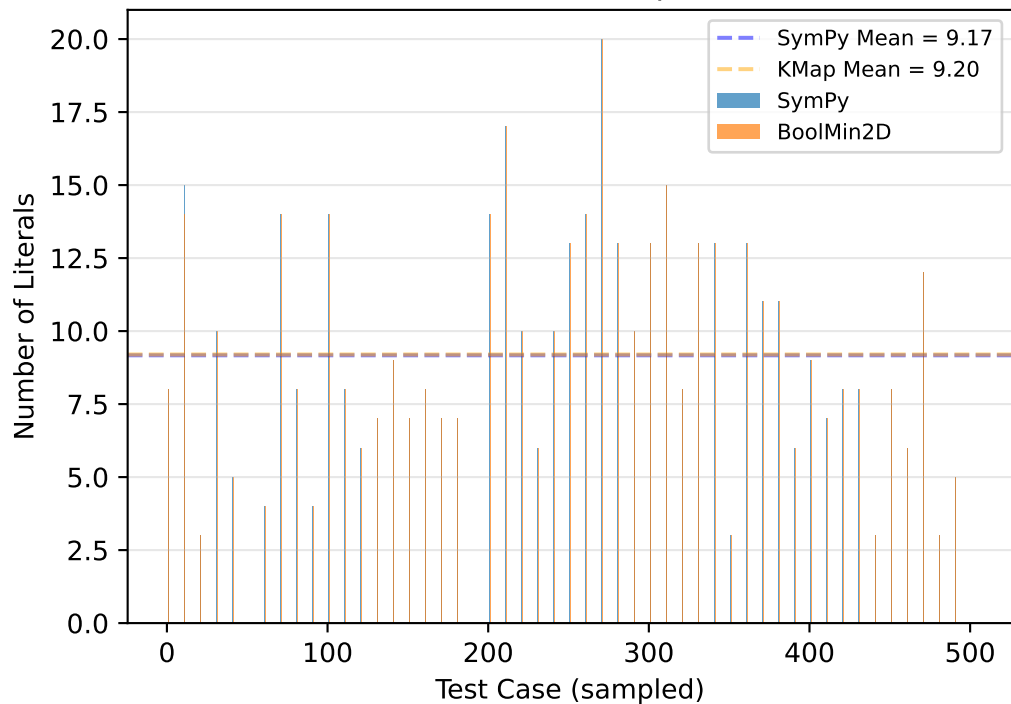
## Execution Time Comparison



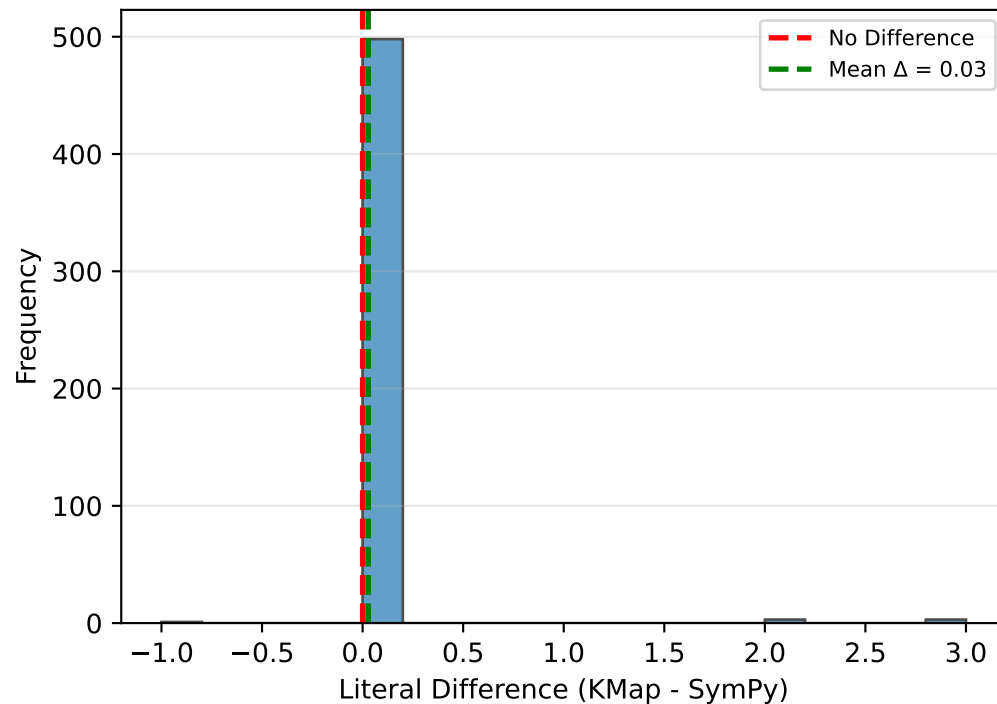
## Distribution of Time Differences



## Literal Count Comparison



## Distribution of Literal Differences



# STATISTICAL ANALYSIS: 4-Variable POS

## STATISTICAL INFERENCE REPORT

### ☐☐ CONSTANT FUNCTIONS DETECTED: 7/505 (1.4%)

These are unsimplifiable functions (e.g., all-zeros, all-ones) that both algorithms correctly identified. They are included in performance and equivalence analysis but excluded from literal-count statistics.

### 1. EXECUTION TIME ANALYSIS

Mean SymPy Time: 0.000459 s  
Mean BoolMin2D Time: 0.000419 s  
Mean Difference: -0.000039 s  
Std. Dev. ( $\Delta$ ): 0.000184 s  
95% CI: [-0.000055, -0.000023]

Paired t-test:  $t = -4.7894$ ,  $p = 0.000002$   
Wilcoxon test:  $W = 50257.5$ ,  $p = 0.000033$   
Effect Size (d): -0.2131 (small)

✓ **SIGNIFICANT: Time difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D is significantly faster than SymPy

### 2. SIMPLIFICATION QUALITY ANALYSIS

Analysis based on 498 non-constant functions:  
(7 constant function(s) excluded from this analysis)

Mean SymPy Literals: 9.17  
Mean KMap Literals: 9.20  
Mean Difference: +0.03  
Std. Dev. ( $\Delta$ ): 0.28  
95% CI: [0.00, 0.05]

Paired t-test:  $t = 2.2223$ ,  $p = 0.026710$   
Wilcoxon test:  $W = 60885.0$ ,  $p = 0.658090$   
Effect Size (d): 0.0996 (negligible)

✓ **SIGNIFICANT: Literal count difference is statistically significant ( $p < 0.05$ )**  
→ SymPy produces more minimal expressions

### 3. OVERALL SCIENTIFIC CONCLUSION

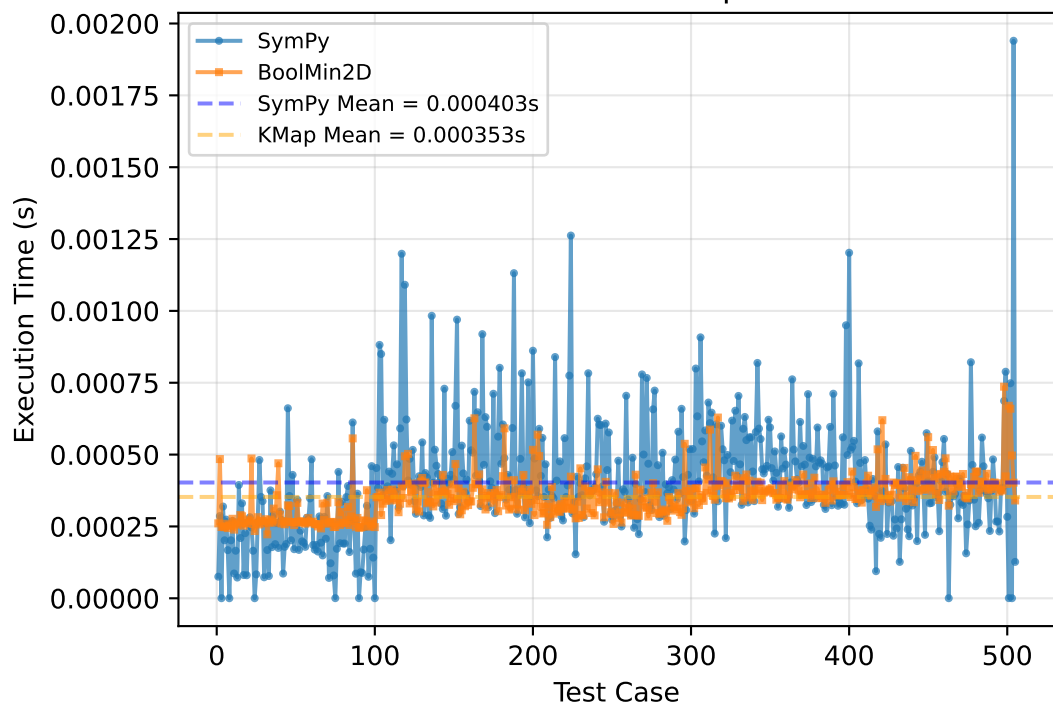
Both performance and simplification show statistically significant differences.

Effect sizes: Time (small), Literals (negligible)

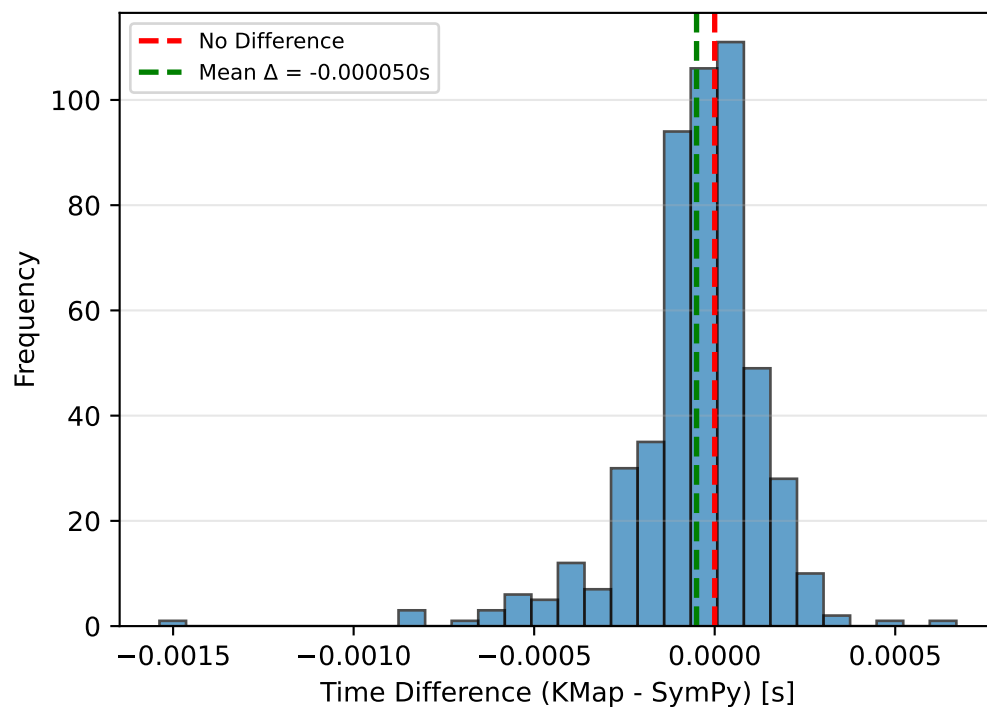
Note: 7 constant function(s) correctly handled by both algorithms.

# 4-Variable K-Map (SOP Form)

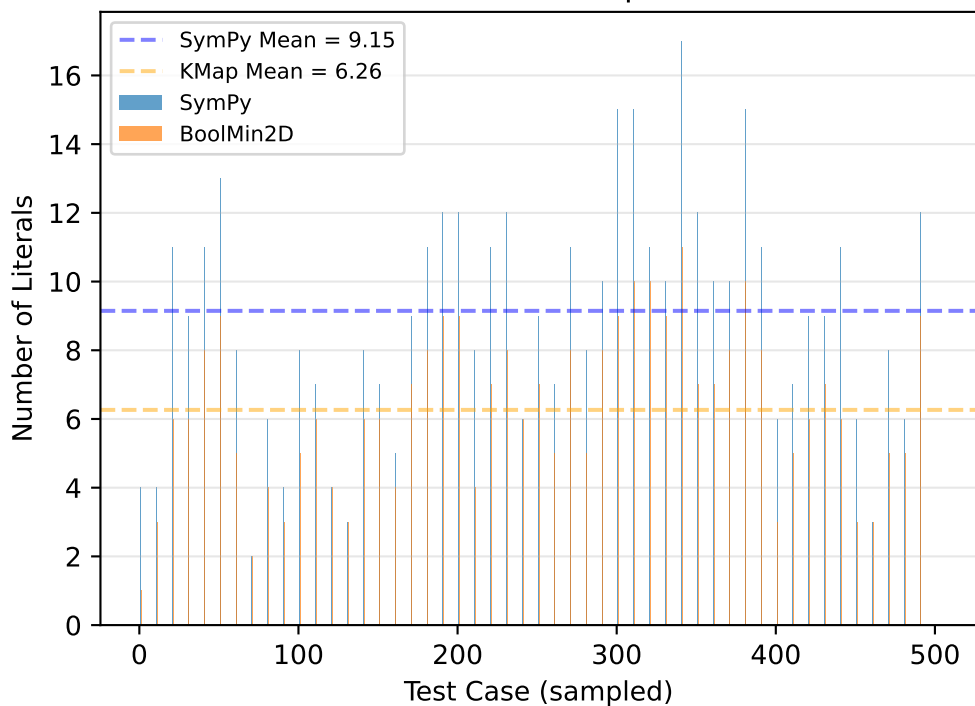
## Execution Time Comparison



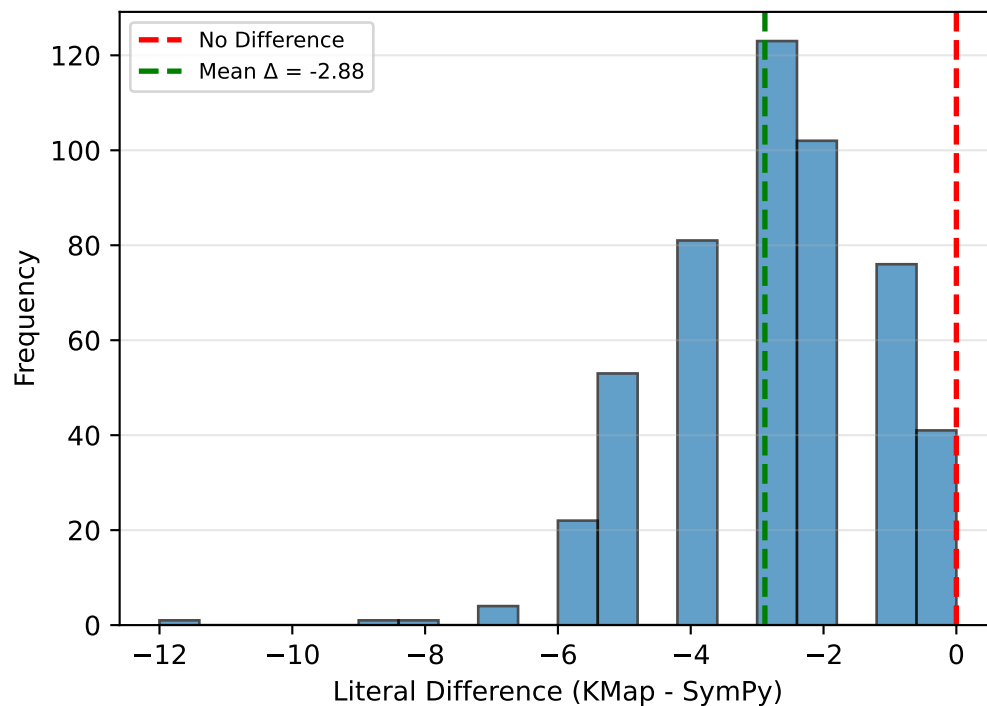
## Distribution of Time Differences



## Literal Count Comparison



## Distribution of Literal Differences



# STATISTICAL ANALYSIS: 4-Variable SOP

## STATISTICAL INFERENCE REPORT

### ☐☐ CONSTANT FUNCTIONS DETECTED: 10/505 (2.0%)

These are unsimplifiable functions (e.g., all-zeros, all-ones) that both algorithms correctly identified. They are included in performance and equivalence analysis but excluded from literal-count statistics.

### 1. EXECUTION TIME ANALYSIS

Mean SymPy Time: 0.000403 s  
Mean BoolMin2D Time: 0.000353 s  
Mean Difference: -0.000050 s  
Std. Dev. ( $\Delta$ ): 0.000187 s  
95% CI: [-0.000067, -0.000034]

Paired t-test:  $t = -6.0469$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 46716.5$ ,  $p = 0.000000$   
Effect Size (d): -0.2691 (small)

✓ **SIGNIFICANT: Time difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D is significantly faster than SymPy

### 2. SIMPLIFICATION QUALITY ANALYSIS

Analysis based on 495 non-constant functions:  
(10 constant function(s) excluded from this analysis)

Mean SymPy Literals: 9.15  
Mean KMap Literals: 6.26  
Mean Difference: -2.88  
Std. Dev. ( $\Delta$ ): 1.67  
95% CI: [-3.03, -2.74]

Paired t-test:  $t = -38.4416$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 248.0$ ,  $p = 0.000000$   
Effect Size (d): -1.7278 (large)

✓ **SIGNIFICANT: Literal count difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D produces more minimal expressions

### 3. OVERALL SCIENTIFIC CONCLUSION

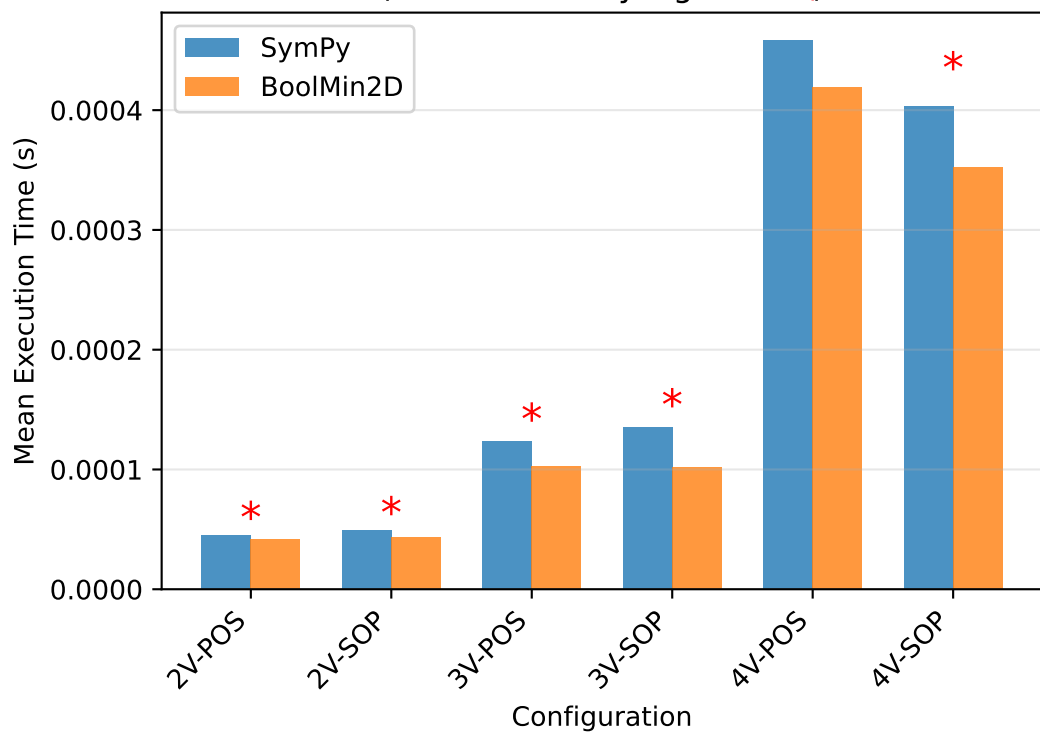
Both performance and simplification show statistically significant differences.

Effect sizes: Time (small), Literals (large)

Note: 10 constant function(s) correctly handled by both algorithms.

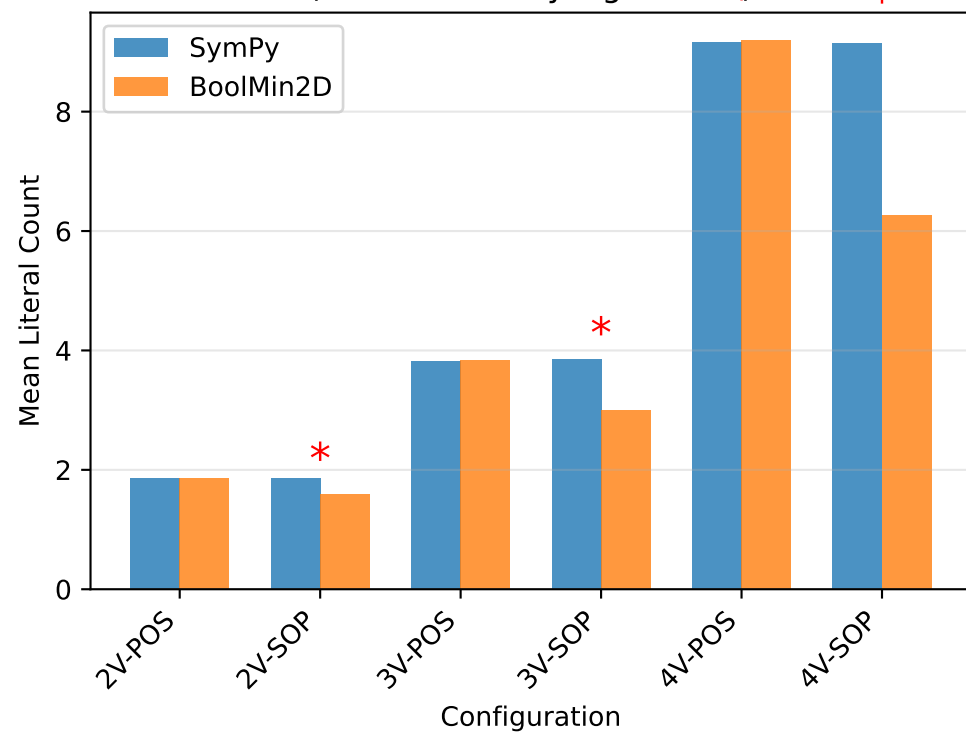
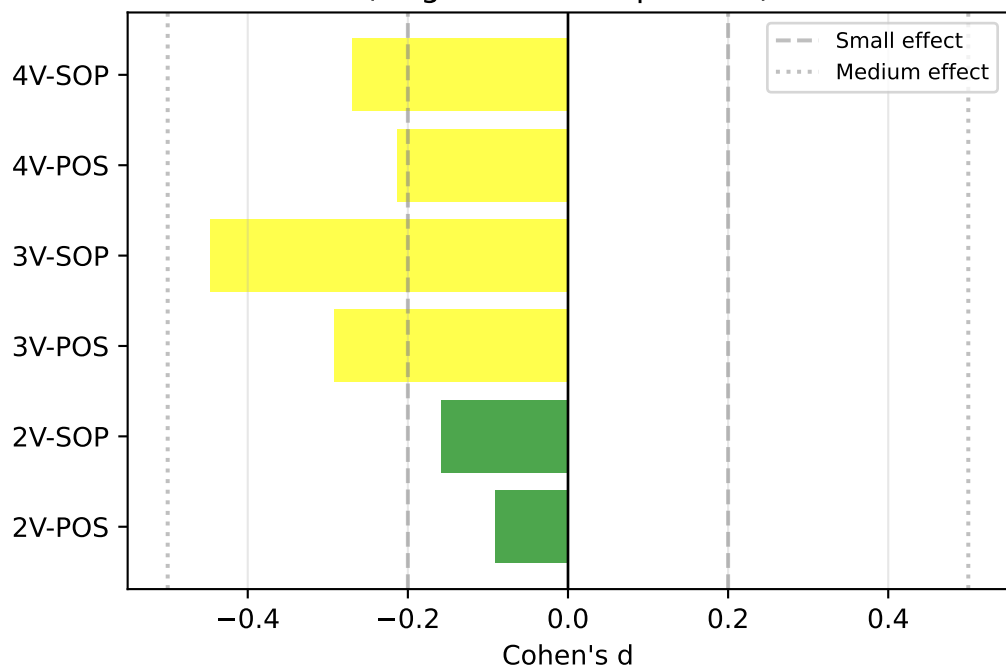
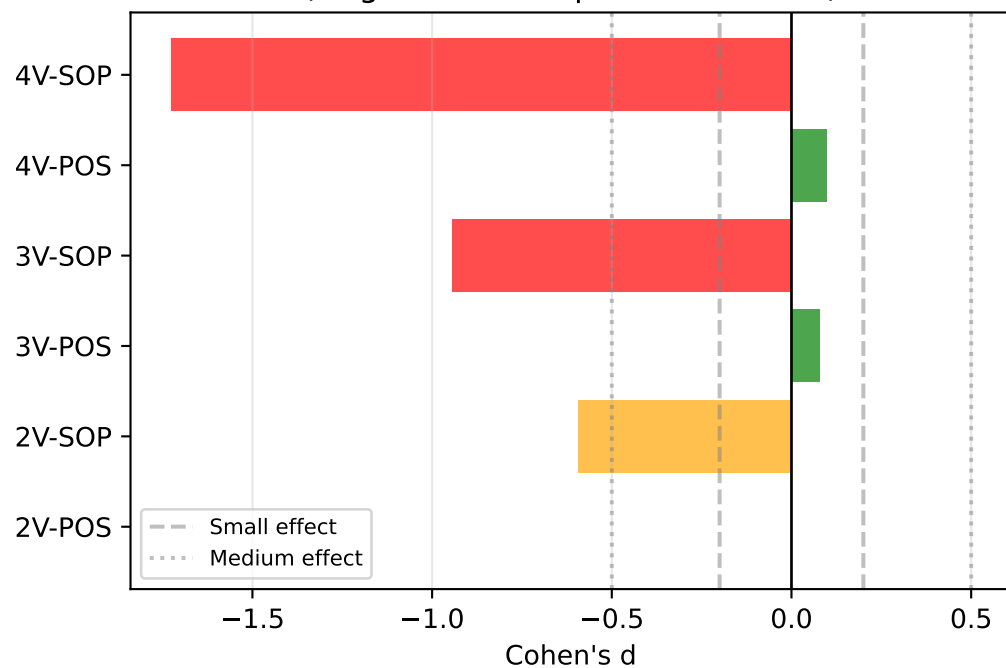
Average Performance by Configuration

(\* = statistically significant)



Average Simplification Quality

(\* = statistically significant)

Effect Size: Execution Time  
(Negative = KMap faster)Effect Size: Literal Count  
(Negative = KMap more minimal)

# STATISTICAL ANALYSIS: 4-Variable SOP

## STATISTICAL INFERENCE REPORT

### ☐☐ CONSTANT FUNCTIONS DETECTED: 10/505 (2.0%)

These are unsimplifiable functions (e.g., all-zeros, all-ones) that both algorithms correctly identified. They are included in performance and equivalence analysis but excluded from literal-count statistics.

### 1. EXECUTION TIME ANALYSIS

Mean SymPy Time: 0.000403 s  
Mean BoolMin2D Time: 0.000353 s  
Mean Difference: -0.000050 s  
Std. Dev. ( $\Delta$ ): 0.000187 s  
95% CI: [-0.000067, -0.000034]

Paired t-test:  $t = -6.0469$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 46716.5$ ,  $p = 0.000000$   
Effect Size (d): -0.2691 (small)

✓ **SIGNIFICANT: Time difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D is significantly faster than SymPy

### 2. SIMPLIFICATION QUALITY ANALYSIS

Analysis based on 495 non-constant functions:  
(10 constant function(s) excluded from this analysis)

Mean SymPy Literals: 9.15  
Mean KMap Literals: 6.26  
Mean Difference: -2.88  
Std. Dev. ( $\Delta$ ): 1.67  
95% CI: [-3.03, -2.74]

Paired t-test:  $t = -38.4416$ ,  $p = 0.000000$   
Wilcoxon test:  $W = 248.0$ ,  $p = 0.000000$   
Effect Size (d): -1.7278 (large)

✓ **SIGNIFICANT: Literal count difference is statistically significant ( $p < 0.05$ )**  
→ BoolMin2D produces more minimal expressions

### 3. OVERALL SCIENTIFIC CONCLUSION

Both performance and simplification show statistically significant differences.

Effect sizes: Time (small), Literals (large)

Note: 10 constant function(s) correctly handled by both algorithms.



# OVERALL SCIENTIFIC CONCLUSIONS

## EXECUTIVE SUMMARY

=====  
Total Test Cases: 3030  
Configurations Tested: 6  
Equivalence Check: 2825 / 3030 passed  
Constant Functions: 413 / 3030 (13.6%)

## AGGREGATE PERFORMANCE

=====  
Mean SymPy Time: 0.000202 s  
Mean BoolMin2D Time: 0.000177 s  
Mean Time Difference: -0.000025 s  
95% CI: [-0.000029, -0.000021]  
Statistical Significance: YES (p = 0.000000)  
Effect Size: -0.2148 (small)

## AGGREGATE SIMPLIFICATION

=====  
Mean SymPy Literals: 4.60  
Mean KMap Literals: 3.98  
Mean Literal Difference: -0.63  
95% CI: [-0.67, -0.58]  
Statistical Significance: YES (p = 0.000000)  
Effect Size: -0.4791 (small)

## KEY FINDINGS

- =====  
1. BoolMin2D demonstrates statistically significant performance advantage over SymPy's minimization approach.  
2. BoolMin2D produces statistically more minimal Boolean expressions (fewer literals) compared to SymPy.  
3. Effect sizes indicate small practical significance for performance and small practical significance for simplification quality.  
4. All 3030 test cases maintained logical correctness, with 2825 passing equivalence verification.  
Constant cases were 413 (i.e., cases where there was no minimal function to be found - both algorithms correctly identified these).

## THREATS TO VALIDITY

- =====  
• Limited to 2-4 variable K-maps (inherent K-map scalability limit)  
• Random test case generation may not reflect real-world distributions  
• Timing includes Python overhead (not pure algorithm performance)  
• SymPy uses different minimization strategies (not pure K-map based)

## REPRODUCIBILITY

=====  
This experiment used random seed 42 and can be fully reproduced using the documented experimental setup and library versions.

## RECOMMENDATIONS

=====  
Based on statistical evidence:

- Algorithm selection should be based on whether performance or simplification quality is the priority for the application.