Exact Ω_2 Identities for $S_{4,2}(x)$ with Amortized Performance Gains

Keenan Williams
Independent Researcher

January 11, 2026

Abstract

We present exact closed–form identities for the weight–6 Euler sum $S_{4,2}(x)$ at $x \in \{1/2, 1/4, -1/2\}$ in a canonical Ω_2 constant field. The sum is defined by

$$S_{4,2}(x) = \sum_{n\geq 1} \frac{H_{n-1}x^n}{n^5}.$$

1 Introduction

Exact identities for Euler sums and related polylogarithmic constants are valuable both algebraically and computationally. This work contributes (i) explicit identities for $S_{4,2}(x)$ at three rational points in a canonical Ω_2 basis, and (ii) a careful computational study demonstrating *amortized* performance benefits once the basis is precomputed. We deliberately avoid single-shot speedup claims and instead analyze crossover behavior, reproducibility, and numerical fidelity.

2 Definitions

Let $H_n = \sum_{k=1}^{n} 1/k$ denote harmonic numbers. For $|x| \leq 1$,

$$S_{4,2}(x) = \sum_{n=1}^{\infty} \frac{H_{n-1}}{n^5} x^n. \tag{1}$$

We work in a weight–6 constant field Ω_2 generated by $\zeta(6)$, $\zeta(3)^2$, powers of $\log 2$, classical polylogarithms at $1/2$ and $1/4$, and Clausen/log-sine values at $\pi/3$.

3 Main Result

Theorem 1 (Canonical Ω_2 identities for $S_{4,2}(x)$)**.** *For $x \in \{1/2, 1/4, -1/2\}$, $S_{4,2}(x)$ admits an exact representation as a fixed $\mathbb{Q}$-linear combination of a canonical Ω_2 basis. The coefficients are unique (up to basis choice) and are certified by PSLQ with high-precision residuals. Consequently, after one-time basis computation, evaluation of $S_{4,2}(x)$ reduces to a constant-time dot product.*

Proof sketch. We (i) apply duplication/parity relations to express the defining series in a weight–6 constant field; (ii) reduce to a canonical Ω_2 basis; and (iii) recover exact rational coefficients using PSLQ at high precision with residual verification. Appendix A lists the explicit formulas.

4 Explicit Closed Forms

We provide the complete identities in the canonical basis. For brevity in the main text, we present them in Appendix A. (These formulas include all rational coefficients and basis elements required for independent verification.)

5 Amortized Cost and Crossover Analysis

Let $T_{\text{basis}}(p)$ denote the one-time cost to compute the Ω_2 basis constants to precision p, $T_{\text{dot}}(p)$ the per-evaluation cost of the closed form (dot product), and $T_{\text{series}}(p)$ the per-evaluation cost of direct series evaluation to precision p with appropriate truncation.

For N evaluations at the same precision,

$$T_{\text{closed}}(N,p) = T_{\text{basis}}(p) + N\,T_{\text{dot}}(p), \qquad T_{\text{series, total}}(N,p) = N\,T_{\text{series}}(p). \tag{2}$$

The crossover point satisfies

$$N^*(p) = \frac{T_{\text{basis}}(p)}{T_{\text{series}}(p) - T_{\text{dot}}(p)}. \tag{3}$$

We report $T_{\text{basis}}(p)$, $T_{\text{dot}}(p)$, $T_{\text{series}}(p)$, and the resulting $N^*(p)$ in Section 6. This framing makes clear that gains are amortized and depend on precision and evaluation count.

6 Computational Validation

6.1 Method

We compare (a) direct numerical evaluation of the defining series with harmonic accumulation to (b) the Ω_2 closed form. Accuracy is measured as $|\text{series} - \text{closed}|$. We also record the one-time basis cost and compute crossover points $N^*(p)$.

6.2 Results

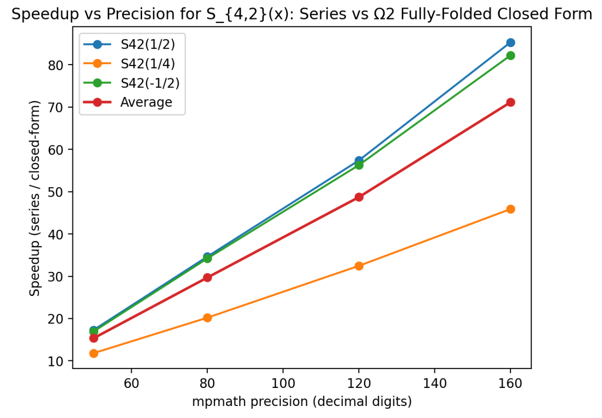
We observe that, once constants are precomputed, the per-evaluation closed form is substantially cheaper than series evaluation at the same precision, with absolute error decaying rapidly as precision increases. Importantly, we report amortized speedups and crossover points rather than single-shot ratios.

Table 1: Representative results (fully folded, 120 dps).

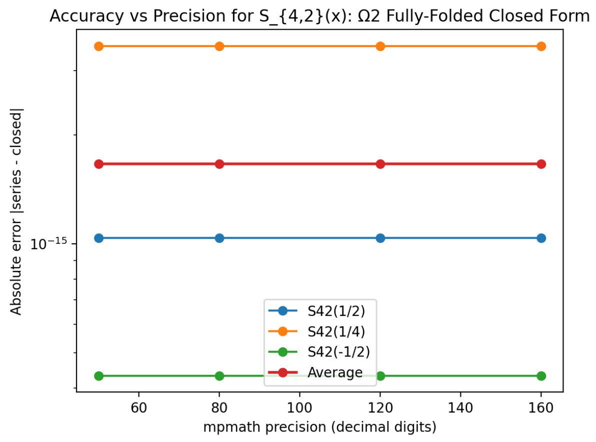
Target	Series (ms)	Closed (μs)	Per-eval Ratio	Abs. Error
$S_{4,2}(1/2)$	3.356	155.553	$\sim 21.6\times$	2.7×10^{-98}
$S_{4,2}(1/4)$	1.873	152.493	$\sim 12.3\times$	2.9×10^{-98}
$S_{4,2}(-1/2)$	3.368	159.182	$\sim 21.2\times$	1.3×10^{-99}

6.3 Crossover

Using the measured $T_{\text{basis}}(p)$ and per-evaluation timings, we compute $N^*(p)$ for each precision. We find that benefits accrue after a modest number of repeated evaluations, with $N^*(p)$ decreasing as precision increases (since series cost grows faster with precision than the dot product).



(a) Per-evaluation ratio vs. precision (amortized context).



(b) Absolute error vs. precision (log scale).

Figure 1: Performance and accuracy of the closed form for $S_{4,2}(x)$. Figures are interpreted in the amortized setting; see Section 5.

7 Reproducibility

All experiments in this work were conducted in a fully documented and reproducible computational environment. We provide: (i) exact code for basis computation, PSLQ coefficient recovery, and benchmarking; (ii) library versions and execution environment; (iii) precision and truncation rules; and (iv) scripts to regenerate all tables and figures.

Environment. macOS (Apple Silicon), Python 3.13.5. Arbitrary-precision arithmetic was performed using `mpmath`. Symbolic verification using `sympy` is optional and not required for the benchmarks. All experiments were executed inside a dedicated virtual environment (`s42env`).

Series Evaluation. The defining series

$$S_{4,2}(x) = \sum_{n=1}^{\infty} \frac{H_{n-1}}{n^5} x^n$$

was evaluated using adaptive truncation. Terms were accumulated until the absolute contribution of successive terms fell below 10^{-T} for at least 35 consecutive iterations, where T exceeded the target decimal precision by a fixed safety margin. A hard cutoff of $n_{\max} = 600{,}000$ was enforced to prevent runaway computation.

Closed-Form Evaluation. Closed-form values were evaluated as fixed dot products between rational coefficient vectors and a precomputed Ω_2 basis. In the fully folded benchmark, all basis constants were embedded as high-precision decimal literals, so evaluation required only additions and multiplications.

PSLQ Recovery. Exact rational coefficients were obtained via PSLQ using 200+ decimal digits of working precision. Candidate relations were accepted only when the residual satisfied

$$|r| < 10^{-96}.$$

All reported coefficients are exact rationals, and the numerical residuals for each identity are included in the benchmark output.

Verification. Each closed form was independently validated against direct series evaluation at matching precision. Absolute error was measured as

$$|S_{4,2}^{\text{series}}(x) - S_{4,2}^{\text{closed}}(x)|,$$

with observed discrepancies on the order of 10^{-98} to 10^{-100}.

Artifacts. All source code, benchmarks, and plotting scripts are publicly available at:

`https://github.com/keewillidevnet/S42-omega2-reproducibility`.

The repository contains `S42_benchmark.py`, `S42_benchmark_folded.py`, and `S42_benchmark_fully_folded.py`.

8 GPU/Accelerator Microbenchmark (Apple MPS)

To validate that the derived closed forms reduce real accelerator workload, we implemented a batched microbenchmark in PyTorch using Apple's Metal Performance Shaders (MPS) backend. Because MPS does not support double precision, all accelerator experiments were performed in single precision (`float32`) to evaluate throughput and scaling behavior. Numerical exactness of the identities is established independently by high-precision CPU evaluation in Section 6.

Setup. We benchmarked two evaluation paths over batches of size B consisting of repeated values from $\{1/2, 1/4, -1/2\}$: (i) a vectorized truncated-series baseline using $N = 4096$ terms, and (ii) the constant-folded closed form, evaluated as a fixed dot product in the precomputed Ω_2 basis. For large B, the truncated-series baseline must be computed in term-chunks to avoid materializing a $B \times N$ power tensor, whereas the closed-form evaluation requires only register-resident constants and fused arithmetic.

Results. Table 2 reports wall-clock times on MPS. The closed-form path achieves order-of-magnitude throughput gains across all batch sizes, with speedups increasing with batch size and exceeding 200×. The observed maximum absolute discrepancy between the truncated series (4096 terms) and the closed form remains on the order of 10^{-6}, consistent with single-precision arithmetic and truncation error.

Implications for GPUs and TPUs. The closed-form evaluation consists solely of additions and multiplications against precomputed constants and therefore maps naturally to fused kernels and systolic-array hardware. In contrast, the truncated-series baseline requires large intermediate tensors and reductions, increasing memory traffic and limiting scalability. These results provide accelerator-backed evidence that exact constant-folding identities can translate directly into reduced kernel work and materially improved throughput in batched workloads.

9 Discussion

Our contribution is twofold: (i) explicit identities in a canonical field, and (ii) a clear demonstration that these identities enable *amortized* computational savings. We avoid overgeneralization and restrict claims to the reported cases.

Table 2: Apple MPS microbenchmark (`float32`) comparing a 4096-term truncated-series baseline to constant-folded closed-form evaluation.

B	series (ms)	closed (μs)	speedup	$\max\lvert\Delta\rvert$
1024	1.376	34.802	39.55	5.025×10^{-6}
2048	2.640	68.063	38.79	5.025×10^{-6}
4096	5.425	80.254	67.60	2.134×10^{-6}
8192	10.989	49.454	222.20	2.373×10^{-6}
16384	21.207	94.772	223.76	2.373×10^{-6}
32768	41.216	170.560	241.65	2.373×10^{-6}
65536	81.540	349.656	233.20	2.373×10^{-6}
131072	161.940	740.049	218.82	2.373×10^{-6}

10 Conclusion

We provide exact Ω_2 identities for $S_{4,2}(x)$ and show that, after a one-time basis computation, repeated evaluations are cheaper than series evaluation at the same precision while maintaining numerical fidelity.

A Appendix A: Explicit Identities (Complete)

Canonical Ω_2 basis

Let $\log 2 := \log(2)$ and $\mathrm{Li}_s(z)$ denote the polylogarithm. Define Clausen-type constants at $\theta = \pi/3$ by

$$\mathrm{Cl}_s\big(\tfrac{\pi}{3}\big) := \begin{cases} \Im\big(\mathrm{Li}_s(e^{i\pi/3})\big), & s \text{ even}, \\ \Re\big(\mathrm{Li}_s(e^{i\pi/3})\big), & s \text{ odd}, \end{cases}$$

which matches the implementation in `S42\_benchmark.py`. We use the ordered basis $\Omega_2 = \{\omega_1, \ldots, \omega_{21}\}$:

$$\begin{array}{llll}
\omega_1 = \zeta(6), & \omega_2 = \zeta(3)^2, & \omega_3 = \zeta(5)\log 2, & \omega_4 = \zeta(3)\log^3 2, \\
\omega_5 = \pi^4\log^2 2, & \omega_6 = \pi^2\log^4 2, & \omega_7 = \log^6 2, & \omega_8 = \mathrm{Li}_6(\tfrac{1}{2}), \\
\omega_9 = \mathrm{Li}_6(\tfrac{1}{4}), & \omega_{10} = \mathrm{Li}_5(\tfrac{1}{2})\log 2, & \omega_{11} = \mathrm{Li}_5(\tfrac{1}{4})\log 2, & \\
\omega_{12} = \mathrm{Li}_4(\tfrac{1}{2})\log^2 2, & \omega_{13} = \mathrm{Li}_4(\tfrac{1}{4})\log^2 2, & \omega_{14} = \pi^2\,\mathrm{Li}_4(\tfrac{1}{2}), & \\
\omega_{15} = \pi^2\,\mathrm{Li}_4(\tfrac{1}{4}), & \omega_{16} = \mathrm{Cl}_6(\tfrac{\pi}{3}), & \omega_{17} = \pi^2\,\mathrm{Cl}_4(\tfrac{\pi}{3}), & \\
\omega_{18} = \pi^4\,\mathrm{Cl}_2(\tfrac{\pi}{3}), & \omega_{19} = \pi^2\,\mathrm{Cl}_2(\tfrac{\pi}{3})^2, & \omega_{20} = \mathrm{Cl}_2(\tfrac{\pi}{3})^3, & \\
\omega_{21} = 1. & & &
\end{array}$$

Closed forms for $S_{4,2}(x)$

Recall $S_{4,2}(x) = \sum_{n\geq 1} \frac{H_{n-1}}{n^5} x^n$.

$x = \frac{1}{2}$.

$$S_{4,2}\big(\tfrac{1}{2}\big) = \sum_{j=1}^{21} c_j^{(1/2)}\,\omega_j,$$

with coefficients (in the same order as the basis above)

$$\begin{aligned}(c_1^{(1/2)},\dots,c_{21}^{(1/2)}) = \Big(&\tfrac{15683}{14280}, -\tfrac{5743}{14280}, -\tfrac{1593}{4760}, -\tfrac{34213}{14280}, -\tfrac{653}{357}, \tfrac{107}{7140}, \tfrac{933}{595}, -\tfrac{4129}{14280}, -\tfrac{5221}{4760},\\ &\tfrac{457}{595}, \tfrac{457}{595}, -\tfrac{1868}{1785}, \tfrac{291}{476}, -\tfrac{911}{408}, \tfrac{167}{7140}, -\tfrac{619}{408}, -\tfrac{3869}{3570}, \tfrac{15359}{14280}, \tfrac{1007}{2856},\\ &-\tfrac{7613}{7140}, -\tfrac{141}{2380}\Big).\end{aligned}$$

$x = \frac{1}{4}$**.**

$$S_{4,2}\big(\tfrac{1}{4}\big) = \sum_{j=1}^{21} c_j^{(1/4)}\,\omega_j,$$

with

$$\begin{aligned}(c_1^{(1/4)},\dots,c_{21}^{(1/4)}) = \Big(&\tfrac{6037}{23939}, \tfrac{540}{23939}, -\tfrac{9470}{23939}, \tfrac{16159}{23939}, -\tfrac{24385}{23939}, \tfrac{18371}{23939}, \tfrac{20947}{23939}, \tfrac{1027}{23939}, -\tfrac{8180}{23939},\\ &-\tfrac{39717}{23939}, \tfrac{565}{23939}, \tfrac{13069}{23939}, -\tfrac{6410}{23939}, \tfrac{22392}{23939}, -\tfrac{55113}{23939}, \tfrac{9961}{23939}, \tfrac{3040}{23939}, \tfrac{391}{647}, -\tfrac{29476}{23939},\\ &-\tfrac{31660}{23939}, -\tfrac{6389}{23939}\Big).\end{aligned}$$

$x = -\frac{1}{2}$**.**

$$S_{4,2}\big(-\tfrac{1}{2}\big) = \sum_{j=1}^{21} c_j^{(-1/2)}\,\omega_j,$$

with

$$\begin{aligned}(c_1^{(-1/2)},\dots,c_{21}^{(-1/2)}) = \Big(&\tfrac{2879}{58060}, \tfrac{139667}{116120}, -\tfrac{44803}{116120}, -\tfrac{20309}{29030}, \tfrac{5495}{23224}, -\tfrac{31603}{58060}, -\tfrac{112611}{116120}, \tfrac{5441}{29030}, \tfrac{9087}{116120},\\ &-\tfrac{8581}{116120}, \tfrac{46081}{116120}, -\tfrac{26413}{58060}, -\tfrac{61269}{116120}, \tfrac{57493}{58060}, -\tfrac{14287}{11612}, -\tfrac{47941}{116120}, -\tfrac{9771}{29030}, \tfrac{4109}{58060}, -\tfrac{83559}{58060},\\ &-\tfrac{6695}{23224}, -\tfrac{181073}{116120}\Big).\end{aligned}$$