An Audited Contract Boundary for COS Caplet Pricing

A Reproducible Lognormal Benchmark with Machine-Checked Scalar Obligations

Dr. Tamás Nagy

tnagyphd@gmail.com

Working Paper • 2026-08-07

Build-std v2.0 | 2026-08-07 18:15 | ff5a5275

Practitioner's Summary

Fast pricing code can agree with a benchmark while its surrounding mathematical narrative remains too broad. This paper studies that distinction for a Fourier-cosine (COS) caplet implementation. Ten Rust test functions pass. Three comparison tests exercise five lognormal caplet cases, one cap assembled from ten caplets, and four Black-style swaptions against reference calculations; seven other tests check narrower properties.

The formal artifact does not prove a complete interest-rate model or a production pricing engine. It checks selected conditional real-algebra obligations: positivity and ordering consequences of supplied discount-factor identities, positivity of supplied scalar error expressions, and elementary parity rearrangements. The model-to-distribution map, the finite COS sum, model-specific convergence, total curve sensitivities, and implementation equivalence remain outside the verified boundary.

The practical contribution is therefore a manually audited contract boundary around a machine-checked scalar proof artifact. It records which inputs would have to be connected before a numerical pricing result could inherit a checked algebraic consequence. This is useful for model governance because it prevents assumptions about distributions, truncation error, or factor reduction from being reported as proved conclusions.

This note does not claim a new COS algorithm, a new interest-rate model, a verified Jamshidian decomposition, or the first formalization of derivative pricing. Its positive evidence is narrower: a reproducible lognormal COS benchmark and a freshly replayed formal proof artifact whose claim surface is explicitly graded.

Abstract

We present a bounded verification case study for COS caplet pricing. The numerical component expands the payoff of a log-forward rate, pairs characteristic-function coefficient proxies with raw payoff integrals, and reproduces Black benchmark values in a lognormal special case. Ten Rust test functions pass. Three comparison tests cover five caplet cases, a ten-caplet semiannual cap, and four swaptions against Black/reference calculations; the remaining tests check narrower numerical properties.

The accompanying formal artifact contains 55 theorem declarations and 13 explicit inputs. A declaration-level audit classifies 7 results as substantive conditional algebra, 30 as arithmetic or definitional checks, 16 as imported aliases, and 2 as tautologies. The artifact replays successfully with zero sorry or admitted proof holes in those declarations, conditional on the 13 explicit inputs. It does not formalize the finite COS series, a model-specific characteristic function, exponential coefficient decay, Jamshidian decomposition, analytical differentiation, or the Rust implementation. We therefore make one main claim: **the combined artifact provides a reproducible numerical benchmark and a manually audited boundary around selected machine-checked scalar consequences.** This is smaller than a verified pricing engine, but it is precise, testable, and suitable as a foundation for stronger future formalization.

1. Problem and Contribution

Interest-rate pricing combines model assumptions, numerical approximation, and implementation details. A benchmark test can validate the final number for one distribution without establishing that a broader factor model generates that distribution. Conversely, a proof assistant can verify an implication while leaving the economically difficult premise as an explicit input. Conflating these levels creates model-risk claims that neither tests nor proofs support.

The COS method is established numerical prior art (Fang and Oosterlee, 2008). Its accuracy depends on the chosen truncation range, coefficient behavior, payoff regularity, and implementation details (Junike and Pankrashkin, 2022; Junike, 2024). Mechanized derivative pricing also predates this work, including a formalization of Cox–Ross–Rubinstein pricing in Isabelle/HOL (Echenim, Guiol, and Peltier, 2020). Coelho (2026) goes further by classifying the faithfulness of formal-finance statements to their mathematical claims. The residual contribution here is only a COS-caplet case study that applies the same broad discipline to a numerical benchmark and its surrounding scalar proof artifact. We claim novelty neither for COS pricing, formal finance, nor claim-faithfulness auditing in general.

This paper contributes a deliberately narrow artifact:

1. a consistent statement of the lognormal COS caplet calculation implemented in Rust;
2. a reproducible benchmark ledger for the checked-in implementation;
3. a declaration-level account of what the formal proof file actually checks; and
4. an explicit list of obligations that remain unverified.

2. Numerical Contract

2.1 State variable and payoff

The contract begins by fixing the random variable and payoff that the implementation actually uses.

Let $L > 0$ be the forward rate under the T_2-forward measure, $X = \log L$, $K > 0$ the strike, $\delta > 0$ the accrual factor, and $P(0, T_2) > 0$ the payment-date discount factor. The caplet value is

$$C = P(0, T_2)\, \delta\, \mathbb{E}\left[(e^X - K)^+\right].$$

The implementation assumes that X is Gaussian in the benchmark case. This is an input model, not a consequence of a multi-factor spectral yield representation.

2.2 Consistent COS convention

Assume $a < b$, positive lognormal volatility, and $0 < K < e^b$. On the truncation interval $[a, b]$, let $u_n = n\pi/(b-a)$. The exact truncated-density coefficient is

$$\widehat{F}_n = \frac{2}{b-a} \int_a^b f_X(x) \cos(u_n(x-a))\, dx.$$

The implementation uses the standard characteristic-function proxy

$$F_n^{\mathrm{CF}} = \frac{2}{b-a} \operatorname{Re}\big(\varphi_X(u_n) e^{-iu_n a}\big)\,.$$

The difference between $\widehat{F}_n$ and F_n^{CF} is an outside-domain approximation obligation; it is not proved away by knowing the characteristic function. The raw payoff integrals are

$$V_n = \int_c^b (e^x - K) \cos(u_n(x-a))\, dx, \qquad c = \max(a, \log K).$$

With the prime assigning weight $1/2$ to the $n = 0$ term, the implemented approximation is

$$C_N = P(0, T_2)\, \delta \sum_{n=0}^{N-1}{}' F_n^{\mathrm{CF}} V_n.$$

The normalization factor appears in F_n^{CF}, not again in V_n. This is the convention used by the Rust benchmark.

2.3 What must be supplied

The numerical result depends on five inputs that are not established by the present formal layer:

- the pricing measure and distribution of X;
- the characteristic function φ_X;
- a justified truncation interval $[a, b]$;
- an expansion length N or an independently justified error tolerance; and
- correct implementation of the coefficient and payoff formulas.

For a general multi-mode yield model, the map from curve factors to the law of a future forward rate is a separate modeling problem. No closed-form characteristic function or universal exponential error rate is asserted here.

3. Reproducible Benchmark

From the crate directory, the command

```
cd rust/risk130
cargo test --lib pricing::cos_caplet -- --nocapture
```

passes all ten test functions. The three comparison tests contain five caplet cases, the assembled cap, and four swaption cases. They compare the COS implementation with the crate's Black-style reference implementation (Black, 1976); the maximum relative error across these ten comparison cases is 8.36×10^{-6}. Selected generated results are shown below.

Contract	COS value	Reference value	Relative error
Caplet: $K = .035, T_1 = 1, T_2 = 1.5, \sigma = .20$	0.000649621767611	0.000649623301200	2.36×10^{-6}
Caplet: $K = .030, T_1 = 2, T_2 = 2.5, \sigma = .15$	0.00218505649958	0.00218505798350	6.79×10^{-7}
Caplet: $K = .050, T_1 = 1, T_2 = 1.5, \sigma = .30$	0.000162989455841	0.000162990817324	8.35×10^{-6}
Ten semiannual caplets	0.0220477432947	0.0220477412411	9.31×10^{-8}
Swaption: $K = .035, T = 1$, 5-year tenor	0.0166122484344	0.0166122324233	9.64×10^{-7}
Swaption: $K = .035, T = 1$, 10-year tenor	0.0592670986258	0.0592670777050	3.53×10^{-7}

The Cargo example emits the numerical payload. A deterministic PowerShell wrapper stored beside the ledger artifacts enriches that payload with the repository revision, runtime metadata, and exact wrapper, generator, and pricing-source hashes. The complete ledger has SHA-256 prefix 22555e4255a10a63; its full hash is bound in the publication passport. Both compared values are computed inside the same research crate, and its Black reference uses an approximate normal-CDF implementation. The table is therefore an internal implementation comparison, not an independent high-precision oracle.

For the baseline caplet, $N = 64, 128, 256$ produces the same value to the displayed precision. This demonstrates numerical stabilization for that case; it does not estimate an exponential convergence rate. The close COS/Black agreement validates the finite-sum implementation under the same lognormal distribution. It does not validate a multi-mode OU approximation or a factor-reduction theorem.

4. Formal Scalar Evidence

4.1 Fresh replay

The formal source is proofs/spectral_ir_pricing/spectral_ir_pricing_proof.py. After updating four obsolete tactic calls to the current kernel behavior, the complete file replays successfully. The repository certification reports 55 theorem declarations, 13 explicit inputs, zero axioms classified as genuine trust debt, and zero admitted gaps.

Structural certification alone is not a paper-claim certificate. A manual grade audit gives the following publication-facing inventory. Its complete declaration lists, grading policy, proof hash, and reproduction commands are recorded in forge/fin_spectral_ir_pricing/verification_ledger.yaml (SHA-256 prefix 9a8cb297e4b874b3; the full hash is bound in the publication passport).

Grade	Count	Interpretation
A	7	substantive conditional real-algebra consequences
C	30	arithmetic or definitional checks
D	2	tautologies; excluded from substantive counts
E	16	aliases to imported bootstrap results; excluded from substantive counts
Total	55	theorem declarations, not 55 substantive pricing theorems

The seven Grade-A results concern discount-factor positivity or ordering under supplied identities and forward-rate positivity under supplied discount ordering. They are useful conditional obligations, but they do not assemble a COS pricing engine.

4.2 Verified and unverified surfaces

The present formal artifact verifies selected scalar implications such as:

- positivity and upper bounds from a supplied simple-interest discount identity;
- ordering of discount factors under supplied scalar conditions;
- forward-rate positivity from supplied positive and ordered discounts; and
- elementary positivity or rearrangement properties for supplied error and parity expressions.

It does **not** presently verify:

- the exponential discount definition used in the numerical model;
- the finite COS sum or its implementation;
- decay of F_n or a model-specific truncation-error bound;
- a multi-factor Jamshidian decomposition;
- analytical total derivatives with respect to yield-curve coefficients;
- cap-floor parity from market primitives; or

- equivalence between the formal artifact and the Rust code.

For this reason, the metadata does not mark the paper as formally verified. The precise positive statement is that a supporting proof artifact replays and checks a bounded set of conditional algebraic obligations.

5. Interpretation and Limitations

The benchmark and the proof artifact answer different questions. The Rust tests show that the implemented COS calculation matches Black values when both use the same lognormal input. The formal file shows that several supplied scalar identities imply expected positivity and ordering properties. Neither evidence source establishes the missing bridge from a general spectral yield model to the benchmark distribution.

The implemented mode sensitivities are frozen-input derivatives: the discount factor, truncation interval, and payoff coefficients are held fixed while the forward input is bumped. They must not be described as total derivatives with respect to curve coefficients. Likewise, the swaption benchmark uses its implemented Black-style input and is not evidence for a formally verified Jamshidian reduction.

The next formalization target is therefore not a larger theorem count. It is one exact end-to-end statement whose objects match the implementation: a finite COS caplet sum with the log-rate payoff, consistent normalization, explicit truncation assumptions, and a generated claim link from the proof source to the paper. Only after that theorem exists should implementation equivalence or model-specific convergence be claimed.

6. Conclusion

This case study converts an over-broad "verified pricing engine" narrative into an auditable contract boundary. The retained positive result is modest but durable: the lognormal COS benchmark is reproducible, the formal artifact replays, and the boundary between assumptions, numerical evidence, and verified algebra is explicit. This makes the draft suitable for further technical development without preserving claims that the current artifacts do not support.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author used Cursor and its integrated AI models (Claude and GPT families) in order to assist with manuscript drafting, structural editing, and language polishing. After using this tool/service, the author reviewed and edited the content as needed and takes full responsibility for the content of the publication.

References

- Black, F. (1976). The pricing of commodity contracts. *Journal of Financial Economics*, 3(1–2), 167–179. DOI: 10.1016/0304-405X(76)90024-6.

- Coelho, Raphael (2026). A Formally Verified Library of Mathematical Finance in Lean 4. *arXiv preprint arXiv:2606.01356*. https://arxiv.org/abs/2606.01356
- Echenim, M., Guiol, H., and Peltier, N. (2020). Formalizing the Cox-Ross-Rubinstein pricing of European derivatives in Isabelle/HOL. *Journal of Automated Reasoning*, 64, 737–765. DOI: 10.1007/s10817-019-09528-w.
- Fang, F. and Oosterlee, C. W. (2008). A novel pricing method for European options based on Fourier-cosine series expansions. *SIAM Journal on Scientific Computing*, 31(2), 826–848. DOI: 10.1137/080718061.
- Junike, G. and Pankrashkin, K. (2022). Precise option pricing by the COS method—How to choose the truncation range. *Applied Mathematics and Computation*, 421, 126935. DOI: 10.1016/j.amc.2022.126935.
- Junike, G. (2024). On the number of terms in the COS method for European option pricing. *Numerische Mathematik*, 156, 533–564. DOI: 10.1007/s00211-024-01402-1.

Appendix A. Reproduction Surface

Formal replay:

```
uv run python proofs/spectral_ir_pricing/spectral_ir_pricing_proof.py
```

Numerical tests:

```
cd rust/risk130
cargo test --lib pricing::cos_caplet -- --nocapture
```

Machine-readable numerical ledger:

```
cd forge/fin_spectral_ir_pricing
powershell -File generate_benchmark_ledger.ps1 -GeneratedOn 2026-08-07
```

The publication passport binds the exact source, PDF, proof, code, generator, and ledger hashes. It remains awaiting exact-hash human approval before external release.