

Pricing SPX Autocalls and Options

with SHTAR

שטר — Symbolic Hierarchical Terms And Rules

Saved Markovian rough-volatility calibration and PyTorch execution

This note shows how to price an SPX memory-coupon autocall and selected European calls and puts from saved Markovian rough-volatility calibrations. It describes one operational and valuation implementation of [SHTAR](#), a **financial product definition language**. Product definitions express economic rights, obligations and conditions through chronological records and dated payments. Event processing interprets those definitions as events occur; the separate model supplies projected market inputs and numerical values. The autocall’s price summarizes its remaining dated obligations; it is not a terminal account balance. The sections below connect the payoff rules, market inputs and regression choices to a worked numerical example.

The numerical example is a *provisional 4,096-path smoke run* saved on 24 October 2026. It combines a synthetic short autocall with two options. The public one-year Citi note is a separate source-derived reference and still requires its own deal-specific inputs before pricing.

1 What the pricer supports

Binding	Current implementation
Root products	SPX memory-coupon autocall and selected European calls/puts
Market inputs	Joint SPX observations; USD carry and settlement unit values
Child payables	USD fixed-currency claims, under an explicit claim-value binding
Event dates	Returned by product transitions; reachable observations and controls bound before simulation
Numerical engine	Existing native Markovian rough-vol simulator, using saved parameters and coefficient segments
Estimator	Joint stock paths, with the explicit two-fold forward-regression policy below
Unsupported routes	Continued child-product MC/PDE, arbitrary module dependency graphs, runtime registration of unbound dates and discretionary valuation

The [language reference](#) defines product identity, replacement states, returned dates and obligations. This table summarizes the capabilities currently available in the SPX adapter.

SHTAR treats currency, shares, bonds and other deliverables uniformly. DIRECT identifies a payable with an accepted liquid-market value basis in the current market-agreement and liquidity context; SYNTHETIC means that basis is absent. Either status may require an analytic model, PDE, Monte Carlo or another supported method. Status can change when the accepted market basis changes; computation method or contract age alone does not determine it.

This profile supports only USD fixed-currency child claims. The example estimates autocall and option values from saved calibration; it does not supply or certify their direct liquid-market price basis or implement dynamic market-status classification. An observed asset price today does not make a future fixing known. See *Exclusive payable routes* and *Numerical leaves* in the language reference.

2 Binding the saved rough-volatility engine

2.1 Joint observations and market units

The current adapter calls the existing native simulator in `sample_stock=True` mode. Let

$$X_t = \exp(\text{snapshot.log_stock}_t).$$

Under its normalized-forward convention the contractual index observation is

$$S_t = F(0, t)X_t, \tag{1}$$

where $F(0, t)$ is the bound market SPX forward in index points. Equation (1) is a unit conversion using market carry, not a sample-dependent normalization. The note's contractual initial level is a separate term.

The simulator accumulates residual stock noise jointly across selected endpoints. Its conditional forward coordinate alone is insufficient for an autocall fixing. Independent draws from successive marginal distributions would also fail to provide the required common path history.

The mathematical diffusion and its drift/correlation conventions remain defined by the engine. The active model identifier is `symmetric_power_reciprocal_variance_v1`; its existing numerical convention is `conditional_physical_xi_then_equilibrium_rough_zeta_v2`. The equations in this note state the product payoffs and this implementation's observation and estimation conventions; they do not redefine that diffusion.

2.2 Clock, coefficients and continuation

This adapter reconstructs the saved `ExpiryTimeMap`: a calendar-identity base clock, elapsed times in 365-day years and positive fitted interval speeds at the recorded expiration knots. All model dynamics use that existing map. Forwards, discounts and contractual dates retain their external market meaning; a product review does not create a fitted speed. See the language reference's registration section for the general date-binding contract.

The adapter preserves the saved global integration schedule and original expiry endpoints. Additional registered observations use the existing schedule routine. This physical/rough-clock model has no magic clock, and the adapter introduces no magic-time subdivisions.

A saved bootstrap result can contain several coefficient segments. They are replayed from the original calibration origin with their original boundaries. A suffix checkpoint alone cannot justify applying its coefficients to the frozen prefix. A result with a prefix therefore also requires its `bootstrap_state.json`.

Native continuation carries the full process and random-stream state between segments. A conditional-only path bank is not relabeled as a joint stock-path bank; the pricer replays the supported native joint-stock process.

2.3 Dates and claims

The valuation date must match the selected saved calibration. Saved market rows supply exact calendar times, forwards and discounts. Additional fixing dates need explicit bindings; the adapter does not invent interpolation or permit projection beyond the selected fitted horizon.

Two fixed-claim value bindings are implemented:

- **Conditional on obligor performance.** Discount the contractual receipt with the explicitly bound USD settlement discount.
- **Supplied claim unit values.** Use an explicit current value per unit of a claim identified by obligor, guarantor, currency and settlement.

A dated claim is directly convertible here only under one of those explicit bindings. The first produces a value conditional on performance, not an issuer-credit-adjusted note value; neither treats an issuer bond as risk-free cash. Ordinary-event scope is explicit. Historical survival and coupon memory must be supplied for a seasoned position. Previously generated but unsettled claims, including past-due unpaid claims, remain separate obligations: a passed settlement date does not establish payment. This prototype has no input ledger for such pre-existing claims, which require separate treatment. Settled receipts remain historical records and must not be delivered again or entered as coupon memory.

Observed quotes and model-produced context values retain their source, dependencies and market basis. Computing or storing a model value does not by itself establish either market status. The current adapter checks supplied binding fields, not a full context-provenance graph or market acceptance; its local `direct` route identifies fixed-claim conversion and does not certify liquidity.

3 Ordinary SPX payoff definitions

3.1 Memory-coupon autocall

For one note, let N be principal, $S_\star$ the contractual initial SPX level, t_k review dates and s_k settlement dates. Let c_k be the coupon amount, B_k^c the absolute coupon barrier, B_k^a an optional call barrier, and B^p the terminal principal barrier.

With unreleased contractual coupon memory m_{k-1} , define

$$A_k = m_{k-1} + c_k, \quad (2)$$

$$C_k = A_k \mathbf{1}_{\{S_{t_k} \geq B_k^c\}}, \quad (3)$$

$$m_k = A_k \mathbf{1}_{\{S_{t_k} < B_k^c\}}. \quad (4)$$

This memory records coupons whose contractual release condition has not been met. It excludes settled receipts and claims already due but unpaid; the expiry rule below applies only to this contingent memory. The aggregate m is an explicit state-representation choice in this product definition. A coupon list would likewise be declared by the product definition; the pricer or interpreter does not choose between those representations.

Coupon processing precedes redemption. At an eligible interim review,

$$H_k = \mathbf{1}_{\{S_{t_k} \geq B_k^a\}}.$$

A called path delivers $N + C_k$, settling at s_k , and terminates. A surviving path delivers C_k , retains m_k , and returns the next review date. If no call is eligible, $H_k = 0$.

The final review is maturity, with no separate automatic-call branch:

$$R_K = C_K + \begin{cases} N, & S_{t_K} \geq B^p, \\ N S_{t_K} / S_\star, & S_{t_K} < B^p. \end{cases} \quad (5)$$

Unreleased coupon memory expires at termination. All coupon, call and terminal comparisons in this executable definition are inclusive. An explicit quantity multiplies the receipts. Continuous barrier monitoring and discretionary exercise are not part of this payoff.

3.2 Selected European options

For right $\eta \in \{+1, -1\}$, strike K , expiry T , settlement s and explicit quantity q , the delivered amount is

$$q [\eta (S_T - K)]^+.$$

Calls and puts can be selected individually in the same request as an autocall. They share model paths and controls. Contract multipliers are explicit quantities; the pricer does not infer an exchange multiplier from the product name.

3.3 Source-derived reference: Citi 17332Y6D5

The local translation records the final public pricing supplement dated 1 July 2026, available from the [SEC issuer filing](#). Its ordinary economics map to equations (4)–(5).

Term	Source-derived value
Underlying	SPX price index only
Issuer / guarantor	Citigroup Global Markets Holdings Inc. / Citigroup Inc.
Principal / initial level	USD 1,000 / 7,499.36
Issue / maturity	7 July 2026 / 7 July 2027; original term 365 days
Coupon	USD 7.50 per monthly review, with memory
Coupon and principal barrier	Printed absolute level 6,786.921; nominal 90.50%
Automatic call	SPX close $\geq 7,499.36$, from the third review on 1 October 2026 through the penultimate review
Settlement	Third contractual business day after interim review; final payment on 7 July 2027, subject to contractual postponement

The twelve review dates run from 3 August 2026 to 1 July 2027. The first two reviews cannot call; the final review is maturity. The printed barrier is retained exactly:

$$7,499.36 \times 0.9050 = 6,786.92080 \neq 6,786.921.$$

For example, final SPX equal to 6,786.921 releases principal and earned memory; the slightly lower nominal-percentage product does not satisfy the printed barrier comparison.

The binding helper accepts settlement dates and an explicit current review index and unpaid amount. Its existence does not establish historical survival. The public note's July 2027 horizon, seasoned state and contractual calendar remain unresolved for the short calibration used in the numerical example.

4 Concrete regression and diagnostic choices

The [language reference](#) describes generic control variates and hedge interpretation. The SPX pricer uses these concrete settings:

- Controls are discounted SPX forward payoffs at the explicitly selected control dates: $C_j = D(0, t_j)[S_{t_j} - F(0, t_j)]$. The forward and discount come from the bound market data and the theoretical control expectation is zero.
- Original path-ID parity determines two folds. Each fold uses OLS slopes fitted on the other fold, with an intercept. Controls remain defined on the full model population after product redemption.
- The implementation centers training columns for the intercept fit, scales nonconstant columns and solves by SVD. Slopes are transformed back into original control units. A rank-deficient hedge decomposition is minimum-norm in the scaled coordinates and need not be unique in original units.
- The output retains both fold hedge vectors, raw/adjusted PV and MC SE, variance ratio, fold sizes and ranks. Reported hedge units are index units per explicit product quantity; the control already includes its discount factor.
- The residual MC standard error is asymptotic. Raw full-population forward gaps and matching selected call-put parity errors are retained separately. This run does not establish time-step convergence.

The contract processor receives unadjusted observations; regression does not alter event comparisons, product state, stock paths or raw payoff values. Generic hedge interpretation is covered by the language reference; the concrete autocall coefficients are reported with the saved results below.

5 Saved provisional results

5.1 Experiment definition

The saved run uses valuation date **5 October 2026**, 4,096 native paths, seed 202611001, and AMD/ROCm. Each regression fold trains on 2,048 paths and values the other 2,048. Forward controls mature on 6 and 7 October.

The selected saved calibration extends through 3 November 2026. Its confirmation used 524,288 paths, a conditional European payoff estimator and the earlier fixed-control policy. The present pricing run uses native joint stock paths and the requested two-fold forward regression. These two path counts and estimator policies refer to different tasks.

The synthetic autocall has principal 1,000, initial level 7,776.709224646587, coupon 7.50, and absolute coupon/terminal barrier 7,000. It can call on 6 October at its initial level and matures on 7 October; both receipts settle on 7 October. The call and put each have strike 7,800, expiry/settlement 7 October and quantity one. The portfolio contains one of each position.

The run assumes ordinary events and claim valuation conditional on obligor performance. It is a smoke test of the implementation and is not a valuation of the public Citi note or a final convergence study.

5.2 Values and Monte Carlo errors

Position	Raw PV	Raw SE	Regression PV	Regression SE	VR
Synthetic autocall	1010.834332	0.058514	1010.804060	0.037445	2.442
SPX call 7800	12.785726	0.395360	12.825688	0.275280	2.063
SPX put 7800	35.486517	0.712143	35.407865	0.275280	6.692
Portfolio	1059.106576	0.684626	1059.037613	0.552451	1.536

Values and errors are in USD present-value units for the explicit quantities. USD is the reporting unit and also the contractual settlement asset in this profile; changing reporting units would not change what is delivered or when. Adding dated claim values neither redelivers settled payments nor assumes their reinvestment. A terminal wealth calculation would require an explicit holding, reinvestment or financing strategy.

The synthetic autocall's sampling standard error decreases from about 0.0585 to 0.0374 in this saved run.

5.3 Autocall forward exposures

Valuation fold	$\hat{\beta}_{6 \text{ Oct}}$	$\hat{\beta}_{7 \text{ Oct}}$
0; trained on fold 1	−0.06746911	−0.00161273
1; trained on fold 0	−0.06891371	−0.00190610

Opposite positive positions in these forwards offset the fitted negative autocall exposures. Fold differences show sampling variation in the hedge estimates.

5.4 Unadjusted forward and parity checks

Diagnostic	Raw error	MC standard error
6 October forward; index points	−0.450433	0.642765
7 October forward; index points	−0.118640	0.940985
7 October call–put parity; USD PV	−0.118613	0.940774

6 Running the example and interpreting its checks

6.1 Request and execution

The supplied `pricing_request.example.json` selects the three positions above. Its key request fields are:

```
{
  "format": "sdl-pricing-0.1",
  "valuation_date": "2026-10-05",
  "claim_basis": "conditional_on_obligor_performance",
  "event_scope": "ordinary_events_only",
  "control_dates": ["2026-10-06", "2026-10-07"],
  "numerics": {
    "num_paths": 4096, "device": "cuda:0", "provisional": true
  },
  "positions": [ ... explicit product definitions and states ... ]
}
```

The last line above is a schematic placeholder; use the supplied file for valid executable JSON.

From the SHTAR project directory, after activating the existing ROCm environment:

```
python pricer.py \
  --calibration /absolute/path/current_best.json \
  --request pricing_request.example.json \
  --output /tmp/sdl-prices.json
```

The loader also accepts `final_pricing_result.json`. Add `--bootstrap-state` when the selected result has frozen coefficient segments. Existing output files are not overwritten.

Requests can explicitly override path count, seed and device, with a provisional label for smaller runs. Final pricing requires at least 2^{20} paths under the workspace rules. The separate fitting and confirmation cap remains 2^{19} ; this pricer does not change it.

6.2 Checks supporting the example

All 24 tests passed in the implementation validation. Checks include all seven source-derived Citi boundary cases, printed-barrier rounding, coupon memory, delayed settlement, early redemption, selected call/put parity, held-out regression fitting and collinear controls.

Checks also require observations generated in successive intervals to match a single simulation with the same parameters, grid and random streams, including continuation across coefficient segments. Missing claim values, inconsistent valuation dates and requests beyond the fitted horizon are rejected.

The saved 4,096-path AMD/ROCM results remain provisional; these checks do not establish final numerical accuracy or time-step convergence.

7 Remaining scope

The current implementation is sufficient for ordinary, explicitly bound SPX deals within a compatible saved calibration horizon. Capabilities beyond this application profile are described in the language reference: module dependency graphs, dynamic registration, continued product valuation and decision handling. They are not claimed as implemented by this pricer.

For the public Citi reference, the next deal-specific inputs are a compatible calibrated horizon through its remaining reviews, verified historical survival and coupon memory, accepted fixing/settlement conventions and a selected claim valuation basis. These inputs determine whether that actual note can be priced.

Acknowledgements. SHTAR — concept and design: Gregory Pelts.