Glidepath Lab: Technical Note

lazyeconomist.com · Methodology series, Paper 2 of 3 · September 2026 · engine v0.4

Engine, API, the browser workbench and the six-factor memo generator — which builds a versioned evidence document from a saved run — are live at glidepath.lazyeconomist.com.

This note documents the equations, defaults and validation tests for each engine in Glidepath Lab. Notation: months are indexed by t; sleeves by $k \in$ {eq, fi, hy, cash, pe, pc, re, ra}; cohorts by c; Monte Carlo paths by ω. All rates are monthly unless stated.

---

1. Lifecycle engine

1.1 Cohorts and contributions

A cohort c is defined by entry age a_0, headcount n_c, starting salary S_0, nominal salary growth g (default 3%), employee contribution rate κ, employer match m and a tenure-dependent separation hazard $h(\tau)$. Salary at month t is $S_t = S_0(1+g)^{t/12}$. An inflation assumption is accepted as a CMA input but is not used by the v0.3 engine — it is stored for a future real-terms feature and has no effect on this run. Monthly contribution per *active* (still-contributing) participant is $(\kappa+m)\cdot S_t/12$.

Headcount and separations are deterministic expected-value paths, not random draws: every cohort loses the same expected fraction to separation each month, given by the monthly hazard implied by $h(\tau)$. Only returns are simulated per Monte Carlo path. Separated participants either roll over (probability ρ, the plan's `rollover_propensity`) or stay in the vintage as terminated-vested balances that continue on the glidepath without further contributions. The engine tracks two counts per cohort: the *active* count, which drives contributions and decays by the full separation rate, and the *headcount with balances*, which only loses the share of separated participants who actually roll out. Per-head figures — the balance used for retirement income — divide by the latter, not by active participants, since a terminated-vested stayer's balance remains pooled in the vintage.

1.2 Public asset returns

Public sleeve returns are drawn from a correlated multivariate lognormal. For annual mean μ and volatility σ, the monthly log-return has mean $\ln(1+\mu)/12 - \sigma^2/24$ and standard deviation

$\sigma/\sqrt{12}$; the engine imposes correlation across sleeves by a Cholesky factorisation of the user-supplied correlation matrix, which it validates for positive definiteness before a run starts. The public sleeves are eq, fi, hy and cash, with default expense ratios of 5, 5, 40 and 5 basis points respectively and default annual μ/σ of eq 8%/16%, fi 5%/6%, hy 6.5%/10% and cash 3%/0.5%. Default correlations are eq–fi 0.10, eq–hy 0.60, fi–hy 0.30. Optional block bootstrap from a user-uploaded history is a v2 feature.

1.3 Private asset returns — economic and reported

For each private sleeve k with funding asset $f(k)$:

Economic return $r^{econ}_{k,t} = r_{f(k),t} + (\lambda_k + \alpha_{k,\omega})/12 + \varepsilon_{k,t}$

where λ_k is the scenario premium (annual), $\alpha_{k,\omega}$ is a per-path, per-sleeve manager alpha drawn once from a triangular distribution `Tri(-3%, 0, +3%)` by default, and $\varepsilon_{k,t} \sim N(0, \sigma^2_{idio}/12)$ is idiosyncratic noise. Economic correlation with the funding asset is therefore inherited from r_f and is not an input; the idiosyncratic term sets how far below 1 it sits.

Reported return (appraisal smoothing) $r^{rep}_{k,t} = \theta_k \cdot r^{econ}_{k,t} + (1-\theta_k) \cdot r^{rep}_{k,t-1}$

This is the standard first-order smoothing filter; θ is the fraction of true return recognised in the current period. The filter is seeded at the drift rate rather than at zero, so the first months of a run are not artificially smoothed toward no return. Under this filter reported volatility is $\sigma_{rep} = \sigma_{econ} \cdot \sqrt{(\theta/(2-\theta))}$ so θ = 0.4 (the PE default) implies reported vol ≈ 0.5× economic — a smaller gap than a naive reading of θ suggests, but still a large one.

Defaults: θ_pe = 0.4, θ_pc = 0.5, θ_re = 0.3, θ_ra = 0.35 — filled in automatically for any private sleeve whose θ is left unset. Idiosyncratic vol (σ_idio, annual) has a general schema default of 10% for any private sleeve; the demo plan and the standard assumption set use pe 10%, pc 5%, re 8%, ra 6%. Real estate (re) and real assets/infrastructure (ra) are fully specified in the demo — CMAs, vehicle terms and fee stacks — but carry a 0% glidepath weight until a user allocates to them; both are funded half from public equity and half from fixed income.

Private credit deferred loss. A user-set economic loss L (default 8% of sleeve NAV) is applied to r^{econ} in the single scenario month t_L, and to r^{rep} in the single month t_L + *defer_months* (default 18) — the whole loss lands at once in reported NAV; there is no spread. (A spread is used elsewhere, for S5's catch-down — see §6.)

1.4 Balance dynamics

The engine tracks two balances in parallel for every private sleeve. The *reported* balance is the one participants transact at: it drives rebalancing, redemption and income conversion, and it

is what a statement would show. The *economic* balance is tracked alongside it as the true value of the assets. The gap between the two — the `stale_mark_gap` — is a first-class, reported output; it is not a side calculation used only for risk.

Fees are applied monthly and annually. The flat portion of the fee stack — management, acquired-fund and platform fees — accrues as 1/12 of its annual rate every month, deducted from both the reported and economic balances. The incentive fee is charged once a year, at year-end, on the sleeve's trailing twelve-month *reported* return (see §4.1); it is deducted from both balances at that point.

1.5 Retirement income

At target age A (default 65) the engine records the reported per-head balance for the cohort. `rollover_propensity` of that balance leaves the plan immediately; the remainder stays in the vintage and, for cash-flow and liquidity purposes, decumulates monthly at the systematic withdrawal rate (default 4%) using the terminal row of the glidepath. Separately, the engine converts the balance recorded at the retirement month into the income figure the model reports, by default through an annuity factor φ (default 16.0). If the plan instead selects the systematic-withdrawal method, income is balance × withdrawal rate. Replacement ratio RR = *income / S_A*. Reported statistics per cohort: the 5th, 25th, 50th, 75th and 95th percentiles and the mean of income, plus *P(income_with < income_without)* and the median stale-mark gap.

---

2. Liquidity engine

2.1 Plan-level cash flow

Each month, per TDF vintage v: $Out_t = separations \cdot \rho \cdot \bar{B} + loans/12 + hardship/12$

Loans and hardship withdrawals are treated as leaving the plan for good — a conservative assumption for liquidity, since it never gives credit for money coming back. After the target date, decumulation replaces separations as the outflow, at the systematic withdrawal rate; there is no separate required-minimum-distribution rule, and there are no `transfers_out`.

Inflows — contributions and merger assets — land in the public sleeves in proportion to the plan's public target weights; private sleeves are only topped up through the rebalancing mechanism below. Outflows are met first from the public sleeves, pro-rata to their current balances (cash is one of those sleeves, so a large cash weight absorbs its share too). Private sleeves are sold, at `forced_sale_discount`, only if the public sleeves are exhausted. A

hypothesis-based property test sweeps outflow rates up to 90% of NAV and private shares up to 50% of NAV, under both cap and notice vehicle terms. It confirms the invariant holds throughout: participant liquidity is always met in full, and total NAV — net of any forced-sale discount incurred — is conserved to the penny.

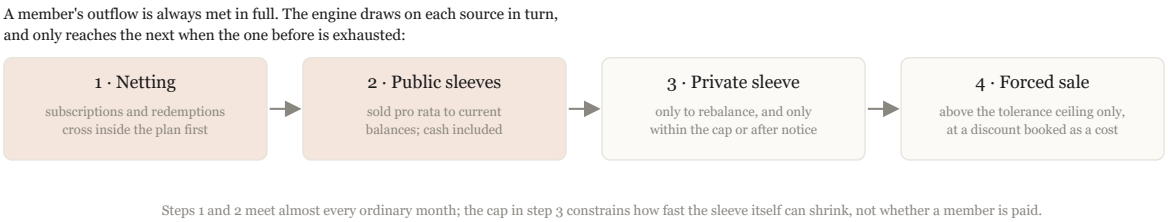


Figure A. The liquidity waterfall. Each source is drawn on only when the one before it is exhausted. The redemption cap sits at step 3 and governs how fast the sleeve itself can shrink; it never stands between a member and her money.

2.2 Rebalancing and the private sleeve

Rebalancing runs on a cadence — monthly or quarterly, default quarterly — and trades a private sleeve toward its glidepath target only when the current weight is outside a band *relative to that target*: the trade fires when weight strays outside `target × (1 ± b)`, with *b* (`rebalance_bands`) defaulting to 5%. (An absolute band would starve a small target sleeve of any rebalancing at all, which is why the band scales with the target.)

Purchases enter a deployment queue and are invested linearly over *D* months (default 12; the demo private-credit sleeve uses 6); capital waiting to deploy earns the cash rate, and the shortfall against the sleeve's own return is booked as cash drag.

Sales compete for the vehicle's redemption cap, which is a fraction of the *whole vehicle's* NAV per cap period (default 5% per quarter; the demo vehicles hold $5bn for private equity and $8bn for private credit). DC subscriptions and redemptions to the same vehicle in one window cross internally first, with no cap and no deployment lag — only the *net* DC redemption after that netting competes for capacity. Other investors in the vehicle request a `base_request_rate` (default 2%) of their NAV per window in the ordinary course. Unfilled other-investor requests carry forward as a vehicle-level backlog that keeps competing with DC requests in later windows — this is what lets a gate outlast the request spike itself. A credit line, sized at `credit_line_pct × NAV` (0% by default; 10% in the demo), extends capacity above the cap when requests exceed it and is repaid out of later net subscriptions; the engine attaches no explicit cost to drawing it.

Diagnostics recorded per redemption window include the fill ratio, the DC backlog and the vehicle-level backlog; a "sustained gate" is flagged once binding windows run *G* deep consecutively (`gate_quarters`, default 3). Per path, the engine reports the longest run of

consecutive binding windows, the number of windows carrying a vehicle backlog, and the peak backlog in dollars.

2.3 Cap terms and notice terms

Vehicle terms come in two modes, selected by `mode`.

	`CAP` terms	`NOTICE` terms (v0.3)
Typical vehicle	US evergreen or interval fund	notice-period fund, or an insurance wrapper holding one
When a request competes	the end of every `cap_period`	the first `dealing_frequency` date after `notice_days` has elapsed
`notice_days`	stored; delays nothing	rounded up to whole months, aged before the request is eligible
`cap_period`	sets the window	ignored
Capacity per window	`cap_pct × NAV`	`cap_pct × NAV`, per dealing date
Fill order	pro rata (every request is the same age)	oldest first, then pro rata within an age
Binding constraint	how much can be sold	when it can be sold, then how much

Under `notice` terms a request ages before it becomes eligible, then competes at the next dealing date under the same cap, netting and other-investor mechanics as before; a notice vehicle with no cap sets `cap_pct` to 1. Fills run oldest first, so a request can never be met ahead of an older one that has also served its notice.

A fund builder (engine module `fund.py`) now simulates an evergreen private-markets vehicle from the inside — pacing, liquidity waterfall, valuation and fees — and writes a catalogue entry; it is a tool under `/app/fund`, not a paper.

Two consequences are worth stating plainly, because neither is visible in the inputs. First, `cap_pct`, `base_request_rate` and `gate_quarters` are all applied *per dealing date* rather than per cap period, so moving a 5% cap from quarterly to monthly dealing triples the annual redemption capacity without a single input changing. Second, because pending subscriptions deploy at each dealing date while sells wait out their notice, the internal netting described above rarely fires under notice terms: a purchase queued this month is deployed before the sale submitted alongside it becomes eligible. Netting is a benefit of same-window settlement, and notice terms remove the same window.

The engine reports two figures in months. Per sleeve, the dollar-weighted months a redemption request waits, computed by Little's law: outstanding request dollars accrue one month at each month's start, and the total is divided by the dollars actually filled. Requests extinguished some other way — trimmed when the balance they were drawn on falls, or swept into a forced sale — accrue waiting time without entering the denominator, so the figure is an estimator of the wait rather than a census of filled requests. A path in which nothing was filled has no wait to report and is left out of the percentiles rather than counted as zero; a sleeve in which no path was filled reports no figure at all. Per cohort, the engine counts the months that cohort's private weight sat above the *top* of its target band. That is a cumulative count of months over the band, not the length of any one spell above it: no run length is tracked for it, and a cohort below the bottom of the band is outside the band and uncounted. What is reported is the worst cohort on each path, read across paths — it is not the plan's own weight (§10). Under cap terms the drift figure that matters is the peak weight a cohort reached; under notice terms it is how many months over the band that cohort accumulated.

2.4 Drift and forced sales

After a public drawdown the private weight rises mechanically because the private sleeve did not fall with it. The engine tracks the running peak private weight per path and cohort. Whenever the private weight exceeds `max_tolerable_exposure` (default 20%), the excess over that threshold is sold immediately — bypassing the redemption cap entirely — at a discount `forced_sale_discount` (default 15%), and the discount is booked as a cost. There is no separate `forced_sale` opt-in flag and no waiting period of G windows before this triggers: it is unconditional and immediate whenever the threshold is breached.

2.5 Buffer location

If `buffer_location = tdf`, no drag is applied to the private sleeve; redemptions are met from public sleeves. If `buffer_location = vehicle`, the sleeve holds a cash fraction β (default 10%) and its return is diluted, applied monthly: $r_k \leftarrow (1-\beta)\cdot r_k + \beta\cdot r_cash$ — equivalent, to first order, to the drag formula the engine actually applies.

2.6 Event scenarios

Bulk transfer: a fraction of the vintage's balance leaves each month for `months` months, met through the ordinary waterfall — the shape of a consolidation seen from the transferring plan, where the existing merger event only adds assets. Layoff: a fraction of active participants separate in a single month, and `rollover_pct` of their balances leave the plan that same month — not spread over several months. Merger: assets equal to a fraction of the vintage's balance are added in a single month; the engine does not support a merger that removes

assets. Switch-to-cash: a fraction of the vintage's balance leaves as an ordinary outflow in a single month, drawn from public sleeves first and then, if needed, from the private sleeve through the redemption cap like any other outflow. Re-enrolment is not modelled in v0.3. Any of these events can be specified directly in the plan's inputs, or supplied by a scenario from the library (§6); both feed the same cash-flow mechanism in §2.1.

---

3. Glidepath specification

The glidepath is a table of sleeve weights by age, interpolated linearly. `shape` is a label on the glidepath, not a behaviour switch: the same interpolation drives both a tapering path and a flat one. `retirement_target` — drawdown, annuity, cash or blended — is not a label: setting it rewrites the terminal (highest-age) row to that target's allocation at validation time, so the change is visible in the stored inputs and the hash rather than applied invisibly at run time. It also selects the income conversion method unless the plan states one explicitly. The templates hold no private weight, on the view that a path ending in one of these shapes has already sold the sleeve. Two shapes are conventional:

- **Taper:** private weights follow user rows, typically higher early and lower near retirement, with a `funding_map` naming which public sleeve each private sleeve is carved from (e.g. PE from equity, PC from high yield, RE from equity and fixed income). The no-sleeve baseline returns the private weight to its funding asset.
- **Constant:** the same table with identical rows at every age, so the private weight (and its composition) never changes with age.

Comparing shapes means running both under the same seed. Every random stream in the engine derives from the run's seed rather than from the glidepath, so two runs seeded identically draw the same public and private returns and pair exactly: the difference in income and in forced-trade and cap-binding cost is attributable to the shape alone. A side-by-side view of that comparison in the workbench is planned but not yet built.

---

4. Fee engine

4.1 Fee stack per sleeve

Annual fee on sleeve NAV: $fee_k = mgmt_k + acquired_k + platform_k + incentive_k$

The incentive fee has no high-water mark. Charged annually, at year-end, on the sleeve's trailing twelve-month reported return r: without catch-up, incentive = $rate \cdot max(r - hurdle, 0)$; with full catch-up enabled, incentive = $min(max(r - hurdle, 0), rate \cdot max(r, 0))$. The choice of reported rather than economic return is deliberate: incentive fees are paid on marks, not on the truth.

Public sleeve fees are flat expense ratios. The TDF total expense ratio at age a is $TER_a = \Sigma_k w_{k,a} \cdot fee_k$, reported with and without the private sleeves, for each premium scenario. The expected private fee used in that calculation evaluates the incentive component at the funding asset's expected return plus the scenario premium.

4.2 Break-even premium

The engine does not solve by bisection. It runs the premium grid — one grid for every private sleeve in the run, taken from the first private sleeve used in the glidepath (§3) — and reads off, for the median cohort by position in the plan, the median income at each premium point. If income is increasing across the grid and the no-sleeve baseline lies between the lowest and highest premium's income, the break-even is found by linear interpolation; otherwise the engine fits a line through the grid and extrapolates. Either way, a result more than one grid span beyond the premiums actually run — or a flat income series that cannot be extrapolated meaningfully — is reported as "not identified" rather than as a number nobody should quote. The break-even is computed and reported for every stress scenario in the run, not only the baseline, and it is net of the sleeve's fee stack; a gross figure is planned but not yet computed.

4.3 TER cap-relevant basis, sleeve tags and allocation constraints

The TER is also reported on a *cap-relevant* basis that omits any fee components the plan tags as `cap_exempt` (a jurisdiction may exempt performance fees from a regulatory fee cap); with nothing exempt the two figures coincide. Plans may attach tag shares to private sleeves (`sleeve_tags`) and declare `allocation_constraints` — a minimum plan-level weight for a set of sleeves, scaled by a tag share, from a calendar year onward (`mc.start_year` anchors month 0). Each constraint is evaluated per projection year on the simulated plan: the median weight and the fraction of paths meeting it.

---

5. Unsmoothing utilities

The engine can estimate θ from a reported return series (`estimate_theta`, an AR(1) regression: $\theta\hat{} = 1 - \rho\hat{}_1$) and de-smooth it (`unsmooth_returns`), but the v0.3 API has no

endpoint that accepts an uploaded history — this is engine functionality waiting on an interface, not a shipped feature.

A correlation haircut multiplier h (default 1.6, result capped at ±0.95) is reported as a valuation diagnostic alongside the simulated reported and economic correlations to the funding asset. It does not feed back into the simulation: the engine's own economic correlation is set by construction, from the funding return plus idiosyncratic noise, not from the haircut.

6. Scenario library parameters

ID	PUBLIC SHOCK	PRIVATE MARK	REQUESTS	FLOWS	OTHER
S1 Baseline	none	none	none	none	—
S2 2022-style drift	eq −25%, fi −15% over 3 months at shock month	reported mark lags economic by 6 months	none	none	drift tracked
S3 BCRED-style gate	none	none	other investors request 9% of their NAV/quarter for 4 quarters vs the 5% cap; DC requests unaffected	none	backlog tracked
S4 Mass layoff	none	none	none	30% separate in one month, 60% roll over immediately	—
S5 Retirement-date stale mark	none	economic mark falls 11.1% one month before the oldest cohort's target date; reported stays 12.5% above economic and catches down evenly over 12 months	none	—	statement vs economic
S6 Australia 2020	none	marks cut 10%, reported lags economic by 6 months	none	10% of balances switch to cash (an outflow) at	fairness transfer (v2)

ID	PUBLIC SHOCK	PRIVATE MARK	REQUESTS	FLOWS	OTHER
				the shock month	
S7 Rate shock + PIK	fi −12%, hy −8% in one month	PC economic loss per the PC CMA (default 8%) recognised in full in reported NAV after `defer_months` (default 18); no spread	none	none	PC economic vs reported

A scenario may also carry **allocation floors**: a minimum combined *target* weight for a set of sleeves from a given month, which raises those sleeves and scales the rest down so each row still sums to one. Floors apply only to the with-sleeve run, since forcing private weight into the no-sleeve baseline would destroy the like-for-like comparison, and they move targets rather than holdings — how long the plan takes to reach the floor is then a liquidity result rather than an assumption.

Premium scenarios {0, +1, +2, +3} pp run orthogonally to S1–S7 — every scenario is run once without the sleeve and once at each premium point.

Switching-fairness transfer (v2 note)

For participants who switch out while reported NAV exceeds economic NAV, the transfer from remaining participants is, conceptually, the gap times the switching balance's private weight and dollar size. This is documented as a planned v2 module; it is not computed by the v0.3 engine.

---

7. Memo generation

The generator maps run outputs to six sections, and the table below shows what each is built from. Every sentence is derived by rule from the run summary, the run's stored inputs, or the four memo-time fields the generator asks for — who the memo is prepared for, whether sleeve returns are stated time-weighted or as an internal rate of return, what participants are told about the sleeve, and any plan context. A memo is therefore reproducible from the run's `inputs_hash` and those fields. Nothing in it asserts a judgement: what the plan record does not carry is named as pending rather than filled in, and each section ends in a committee note the fiduciary writes, under a prompt saying what that note should address.

FACTOR	WHAT THE RUN EXPOSES
Performance	Cohort income percentiles (p5/p25/p50/p75/p95, mean); *P(income with < without)* per cohort
Fees	TER with/without the sleeve, by entry age and premium; the fee-stack table; break-even premium per scenario
Liquidity	Cap-binding probability by year; gate-duration histogram; sustained-gate probability; the dollar-weighted request wait; backlog and vehicle-backlog percentiles; drift-overshoot probability; peak private weight and months over the top of the band, both per cohort (§2.3); forced-sale and cash-drag cost per head
Valuation	θ per sleeve; reported vs economic volatility and correlation to the funding asset; haircut-implied correlation; stale-mark gap (median)
Benchmarking	The objective text and benchmark names are not computed: they are read from the plan record. The engine supplies the allocation being benchmarked — the plan's private weight as a median taken across paths in each year, a year-by-year median that no single path follows
Complexity	Vehicle terms, AUM and investor mix, stored with the plan's inputs; the governance liquidity-action threshold

Every run's summary carries `engine_version` and `inputs_hash`, and governance inputs (objective, benchmarks, revaluation triggers) are stored with the plan. A memo's provenance block carries the run id, that hash, the engine version, the scenarios the run covered, the premium the memo's sentences cite, the time it was generated and its version number. Editing an input and re-running produces a new hash; regenerating a memo writes the next version against the same run rather than overwriting the last. The document is rendered and printed by the browser — there is no server-side rendering of it to a file.

8. Defaults and their sources

PARAMETER	DEFAULT	BASIS
Public equity μ/σ	8% / 16%	provider CMA midpoint
Public fixed income μ/σ	5% / 6%	provider CMA midpoint
Public high yield μ/σ	6.5% / 10%	neutral assumption (not in the underlying spec)
Public cash μ/σ	3% / 0.5%	neutral assumption (not in the underlying spec)
Public expense ratios (eq/fi/hy/cash)	5 / 5 / 40 / 5 bp	neutral assumption
Correlations (eq–fi / eq–hy / fi–hy)	0.10 / 0.60 / 0.30	neutral assumption
Premium grid	0/+1/+2/+3 pp	spans AQR (≈0) to provider (+2–3) estimates
θ (PE/PC/RE/RA)	0.4/0.5/0.3/0.35	de-smoothing literature ranges
σ_idio annual (PE/PC/RE/RA)	10%/5%/8%/6%	illustrative dispersion assumption
Manager alpha	Tri(−3%, 0, +3%)	dispersion evidence; centred on zero
Vehicle terms `mode`	`cap`	the US evergreen/interval convention; `notice` is opt-in
Cap	5%/quarter	standard evergreen/interval-fund term
`notice_days`	60 (inert under `cap` terms)	illustrative; only bites under `notice` terms
`dealing_frequency`	monthly	the common notice-fund term
`cap_exempt`	none	no fee component is exempt unless a jurisdiction's rules

PARAMETER	DEFAULT	BASIS
		say so
`sleeve_tags` / `allocation_constraints`	none	supplied per plan; no constraint is evaluated unless declared
`mc.start_year`	unset	required only when allocation constraints are declared
`plan.earnings_band`	unset	contributions accrue on full salary unless a band is given
`glidepath.retirement_target`	unset	the terminal row is whatever the user enters
Deployment	12 months (PE, RE, RA); 6 months (PC, demo)	evergreen deployment experience
Other-investor base request rate	2%/window	illustrative
Credit line	0% (schema default); 10% in the demo	illustrative
Demo vehicles (AUM)	PE $5bn, PC $8bn, RE $4bn, RA $6bn	illustrative
PE fee stack	125 bp mgmt, 12.5% incentive over 5% hurdle with catch-up, 40 bp acquired, 15 bp platform	semiliquid fee surveys (~3% all-in)
PC fee stack	100 bp mgmt, 10% incentive over 5% hurdle, no catch-up, 20 bp acquired, 15 bp platform	semiliquid fee surveys
RE fee stack	100 bp mgmt, 10% incentive over 5% hurdle, no catch-up, 30 bp acquired, 15 bp platform	open-end core real estate fee surveys
RA fee stack	100 bp mgmt, 10% incentive over 6% hurdle, no catch-up, 30 bp acquired, 15 bp platform	open-end infrastructure fee surveys

PARAMETER	DEFAULT	BASIS
PC deferred loss	8%, recognised in reported NAV after 18 months	illustrative
Annuity factor	16.0	real joint-life at 65, illustrative
Systematic withdrawal rate	4%	illustrative
Demo plan	8 cohorts of 1,200, entry ages 25–60 in 5-year steps, $65,000 starting salary	illustrative
Separation hazard by tenure	18% (0–2 yrs) / 9% (3–9 yrs) / 4% (10+ yrs)	illustrative
Rollover propensity	55%	illustrative
Loan / hardship rate	2%/yr / 0.5%/yr	illustrative

All defaults are editable and are labelled *illustrative* in this note. A memo generated on them is watermarked on every page, and every memo carries a provenance line naming its assumption source — for an edited run, one that says how many series differ from those defaults.

9. Validation

The suite covers the engine, the API and the site pages. Among its tests:

- **Property test (Hypothesis):** across outflow rates up to 90% of NAV and private shares up to 50%, under both cap and notice vehicle terms, participant-level liquidity is always met exactly and total money — net of any forced-sale discount incurred — is conserved.
- **Notice-mode end-to-end run:** a full simulation on 90-day notice terms with monthly dealing produces finite, non-negative balances and a median request wait within a month of the notice period itself.
- **The wait is the notice period rounded up to the dealing calendar,** not the notice period itself: across 90 and 95 days' notice against monthly and quarterly dealing, the median wait is $W\lceil n/W \rceil$ months, where n is whole months of notice and W the months between dealing dates. The case that matters is 95 days with quarterly dealing, where eligibility falls one month after a dealing date and the wait is six months rather than four.

- **Hand-derived dollar tests** for: subscription/redemption netting, cap binding with backlog carry-forward, the credit line's extra capacity and repayment, other-investor crowd-out of DC requests, deployment-queue lag, and forced sale (both the ordinary "public exhausted" case and the drift-driven forced sale above `max_tolerable_exposure`).
- **S3 reproduction:** on every path, the BCRED-style gate scenario produces at least three consecutive gated quarters, with the vehicle backlog and DC fill ratio both showing the sustained gate.
- **A fixed-seed golden file** of a full run's summary output, and a golden copy of the memo document generated from it, both checked on every build — so a changed sentence in a memo is a diff someone has to explain.
- **Serial-vs-parallel identity:** running the same inputs with one worker and with several produces identical summaries.
- **An end-to-end smoke test** that every scenario in the library, S1 through S7, runs without error.
- **A premium sanity check:** with idiosyncratic noise and manager dispersion turned off, a +10 percentage-point premium — well above the ~3% all-in private fee load — beats the no-sleeve baseline on the large majority of paths.
- **A buffer check:** locating the cash buffer inside the vehicle, rather than at the TDF level, measurably lowers median income for the cohort that funds it.

---

10. Limitations

Headcount, salary, contributions and separations are deterministic expected-value paths, not simulated draws; only returns vary by Monte Carlo path. One premium grid is applied to every private sleeve in a run — a plan with more than one private sleeve cannot currently test different premium assumptions for each. Other investors in a vehicle keep subscribing at their base rate even while the vehicle is gated, which is optimistic relative to the 2026 BCRED episode, where other-investor inflows collapsed. Notice periods delay execution only under `notice` terms; forced sales at the tolerance ceiling still bypass both the cap and the notice period, standing in for a secondary sale. The request-wait figure is a Little's-law estimator, so request dollars extinguished by a trim or a forced sale rather than by a fill accrue waiting time that no fill ever offsets. The months-above-band figure is computed per cohort, not for the plan: it counts the months in which a cohort's private weight, aggregated across the private sleeves, sat above that cohort's aggregate target band. The reported figure is the worst cohort on each path, read across paths. It is coarser than the per-sleeve test that actually triggers a rebalance: a cohort overweight one sleeve and underweight another can sit inside the aggregate band while both sleeves trade. The peak private weight beside it is reduced the same way, and neither figure is the plan's own weight, which stays nearer the tolerance because the excess is sold as soon as a cohort breaches it. An inflation assumption is accepted but unused.

Lognormal public returns understate tail dependence (block bootstrap planned). Manager alpha is a single per-path, per-sleeve draw, not time-varying. Drawdown-fund mechanics — capital calls, uncalled commitments, Takahashi–Alexander pacing — are not modelled in the liquidity engine, which sees a vehicle only through its dealing terms. They are modelled inside the fund builder (§2.3), whose output reaches this engine as a catalogue row rather than as mechanics. Taxes and participant-level annuity pricing are ignored. The switching-fairness transfer described in §6 is a v2 module, not yet implemented. A memo cannot name a stored assumption set: there is no link from a run to a saved one, so it compares the run's assumptions against the profile's defaults and reports how many series differ. That is why the absence of a watermark means "not the shipped defaults" rather than "a set someone reviewed". Nothing compares two memos, or two runs, against each other; the workbench lists what has been saved and reopens one at a time. The model does not and cannot resolve the dispute over whether an illiquidity premium exists; it makes the dispute visible.