ATLAS Framework v2.4R2 — 操作辞書 × Temporal Gauge（完全統合＋第2稿レビューバック対応版）

日付: 2025-10-10
互換性: v1.95R5 / v2.0 / v2.0R1（Expert / Grad-Postdoc Companion / Reviewer-Addendum）/ v2.1 / v2.2 / v2.3 / v2.3R1 / v2.4 / v2.4R1 と**後方互換**（定義は不変、キーは上位互換）

今回の更新趣旨（v2.4R2）
第2稿の批判的レビューを受け、**(A)** TG-Ind の**検証可能化**（プロトコル追加）、**(B)** ホロノミー**下界の誤差予算を可計算化**（式と係数・校正手順）、**(C)** **外挿失敗検出の閾値校正**（ROC 必須化・系 **KMS/Tomita – Takesaki 整合**の**実用化**（フォールバック方針と検査アルゴリズム）、**(E)** **API 型定義とエラー **クイックスタート**と**MWE**、**(G)** **計算コスト・スケーリング報告**の定式化、**(H)** **感度分析の具体化**、 の収束規約**を**Normative**に格上げ。
また、レビュー本文に対する**内部批判的レビュー**を実施し、**要求過剰・不必要な一般性・実装負担の偏り**を是正した

---

0. クイックスタート（実装者向け）

1. `ATLAS_thresholds_v2.4R2.json` を読み込む（必要なら `schema_migration` で旧版をアップリフト）。
2. **外部アンカー**を `provenance` に記録：`source_type/id_or_doi/version_or_date/method/ci_95`。
3. **TG-Ind 検証**を §1.3 のプロトコルで実行（結果を JSON Lines に出力）。
4. `run_pipeline(cfg, data)` を実行（§5 の API 型定義に準拠）。
5. 出力 JSON Lines を **バリデータ（付録 I）**で検査（Δ/N/H/SG/Determinism/ROC）。
6. **TPR/FPR/AUC/best-J** と **コスト表**を論文／レポートへ転記。

---

1. 前提と立場（循環回避の実装可能化）

1.1 A0-TG と $\Lambda_\circ$ の時間独立性（再掲）
(TG-Ind): パラメタ多様体 Λ 上の可積分分布 $\mathcal{D} \subset T\Lambda$ を、時間ベクトル と**独立**に構成し、 $X_t \notin \mathcal{D}$,
$[\mathcal{D}, \mathcal{D}] \subseteq \mathcal{D}$。 $\mathcal{D}$ の葉を **$\Lambda_\circ$** とする。

1.2 誘導時間（付録 D）— 存在は局所・仮定依存（ギャップ明記）

1.3 **TG-Ind 検証プロトコル（新・Normative）**

入出力
入力：パラメタ空間記述 `Λ`, 候補時間方向 `X_t`, 保存量／幾何量のセット `I = {I_k}`。
出力：`TG_Ind_Result = {status ∈ PASS|FAIL|UNCERTAIN, D_basis, frobenius_resid, orthogonality, notes}`。

手順（疑似コード）
```python
def verify_TG_Ind(Lambda_desc, X_t, invariants):
    # 1) 候補分布 D の構成：I の共レベル集合の接空間を張る基底 {b_i}
    D_basis = build_tangent_basis(invariants)          # Λ の chart ごとに数値/記号で構成
    # 2) 可積分性（Frobenius）を有限差分で評価：‖[b_i, b_j] - Σ_k c^k_{ij} b_k‖ ≤ τ_Frob
    frob = finite_diff_frobenius(D_basis)
```

```
  # 3) 時間独立性：max_i｜ X_t, b_i ｜≤ τ_orth （計量が無ければ同次座標内積）
  orth = max_inner_product(X_t, D_basis)
  # 4) 判定
  if frob ≤ τ_Frob and orth ≤ τ_orth: status = "PASS"
  elif frob > 10*τ_Frob or orth > 10*τ_orth: status = "FAIL"
  else: status = "UNCERTAIN"
  return TG_Ind_Result(status, D_basis, frob, orth)
```
既定値：`τ_Frob=1e-3`, `τ_orth=1e-6`（HPC: 厳しめ）。閾値は `triage.roc` の校正に従い系別に更新可。
ログ：判定根拠（`frob, orth, invariants, charts`）を JSON Lines へ必記。

---

2. 基礎定義の集約（再掲＋微修正）
- **CPTP 主束・接続・曲率**（Stinespring → Ehresmann 接続）。
- **Δ_chart**：HS 基準／op 換算；`T^†` は **Heisenberg 双対**。
- **N_mod**：幾何（Γ）と熱（KMS）経路。
- **H_top**：辞書接続ホロノミー。

---

3. 規格（Normative）— **下界の誤差予算を可計算化**

定義（保守的下界）
\[
H_{\mathrm{lb}} = \max\Bigl(0, \; H_{\mathrm{obs}}
- E_{\mathrm{disc}}
- E_{\mathrm{loc}}(\Delta)
- E_{\mathrm{resp}}(|\delta N|)\Bigr).
\]

3.1 誤差予算の算出式（新）
- **離散化**：`method="richardson_remainder"`, `order=2`
  \(E_{\mathrm{disc}} = s_f\,\frac{|H(2a)-H(a)|}{(2^p-1)}\), 既定 \(p=2\), 安全率 \(s_f=1.5\)。
- **局所誤差由来**：`functional_form="linear"`
  \(E_{\mathrm{loc}}(\Delta)=c_{\Delta}\,|\Delta|\), 係数 \(c_{\Delta}\) は **校正系**で回帰し、`profiles.*`に格納。
- **応答誤差由来**：`functional_form="linear"`
  \(E_{\mathrm{resp}}(|\delta N|)=c_{N}\,|\delta N|\), 係数 \(c_N\) も同様に校正。
校正手順：校正データで \((|\Delta|,|\delta N|,H_{\mathrm{obs}})\) を収集 → 既知のカーブで **下側回帰**（Qu
τ≈0.1）→ 係数を決定。
運用：係数は JSON に明示；未設定時は **保守的上限**（大きめ係数）を適用。

---

4. 統計・数値安定性

4.1 外挿失敗検出（根拠の付与）
- `N_mod.extrapolation_guard.order_agreement_tol` と `oscillation_max` は、**校正 ROC** により決定。
- **必須出力**：AUC / best-J 閾値 / 95% CI / 係数の系別推奨（spin/boson/fermion）。

4.2 ROC 校正の運用
- **K-fold**（k≥5）で ROC を安定化、**外部検証セット**（例：XXZ/Hubbard）で転移性を評価。
- `triage.report.required=true`（既定）。`triage.roc` に kfold/外部セット DOI を記録。

---

5. 実装 API（型定義・I/O・エラー規約）

```python
from typing import TypedDict, Literal, Any

class DensityMatrix(TypedDict):
    dim: int
    data: list[list[complex]]

class Operator(TypedDict):
    dim: int
    data: list[list[complex]]
    name: str

class SystemState(TypedDict):
    params: dict[str, float]          # 物理パラメタ（J,h,β 等）
    rho: DensityMatrix | None         # 状態（または None）
    channels: dict[str, Any] | None   # CPTP チャネル表現（Kraus 等）
    observables: dict[str, Operator]  # 計測する O

class StageResult(TypedDict):
    stage: Literal["sg","delta","nmod","htop","triage","roc","tg_ind"]
    status: Literal["PASS","WARN","HALT","INCONCLUSIVE","ERROR"]
    metric: str | None
    value: float | None
    threshold: float | None
    aux: dict[str, Any]
    notes: str
    timestamp: str

class AtlasConfig(TypedDict): ...
```

例外と継続方針
- `AtlasConfigError` → **即中断**。
- `AtlasComputationError` → `StageResult{status="ERROR", aux.continue=false}` で停止。
- `RecoverableError` → `status="ERROR", aux.continue=true` で次段へ進む。

並列実行：`determinism.parallel_mode="counter"` を推奨。`aux.thread_id/seed` を必記。

---

6. 計算コストとスケーリング（報告を義務化）

- **理論オーダ**：Δ: \ (O(R d_r^2) \ ), N: \ (O(K B S) \ )（次数×ブートストラップ×サンプル）, H: ループ×解像。
- **実測報告**（必須）：`system_class, N, d_r, bootstrap, loops, resolutions, wall_time_s, env_hash`。
- **スケーリング推定**： \ (T \ \approx \ \alpha N^p B \ ) を**対数回帰**で報告（係数・p の CI を付与）。

---

7. 実証パッケージ（必須レポート）

- **TPR/FPR/AUC/best-J/CI**、**三面ダッシュボード**（Δ/N/H）を必須。
- **外部検証**：転移性を示す ROC を1系以上（XXZ/Hubbard 等）。

---

8. 制限とフォールバック

- **TG-Ind 非確定**：`status=UNCERTAIN` のままでも**運用可**だが、A0-TG の主張は**弱化**。
- **KMS 整合が崩れた場合**：`kms_policy="geometric_only"` にフォールバックし、N_mod は幾何経路のみ（`WARN`

---

9. 物理的解釈と例

9.1 **CPTP 曲率例（振幅減衰 × 位相回転）** — 実装スケッチ
- ループ順序差による出力差を**非可換性（曲率）**の指標として可視化。
- 付録 L に**Python スケッチ**（Qiskit/QuTiP 相当 API）を掲載（実測値は環境に依存）。

9.2 **KMS/Tomita – Takesaki** — 反例とフォールバック
- 反例：局所モジュラー流と接続が可換でない単純模型を提示。
- 方針：`kms_compat_test` が失敗時、**自動的に `geometric_only`** へ降格。

---

10. 内部批判的レビュー（本レビューバックへの所見・要約）

- **TG-Ind の「検証不可能」主張**は妥当だが、**有限差分 Frobenius と直交テスト**で**実務プロトコル**化可能（完璧な
- **下界係数の不明瞭性**の指摘は正当。**定式化＋校正法**を規格に昇格。
- **閾値根拠**は ROC 必須化で解消。過度な系普遍値の強要は避け、**系別プロファイル**で管理。
- **KMS 整合**は理論的に要求が強すぎるため、**フォールバック指針**を用意。
- **API 不備**は型定義・例外規約で補完。
- **ドキュメント膨張**にはクイックスタートで入口を整備。
- **具体例の数値**は再現環境依存であるため、**コードと評価手順**を優先提示。

付録 H: スキーマ移行（更新）
（v2.4R1 と同様、v2.4R2 で追加されたキーを自動補完）

付録 I: 自動バリデータ
P1〜P5 に加え、**P6（TG-Ind）**、**P7（誤差予算の係数設定）**、**P8（ROC 校正の存在）** を追加。

付録 J: KMS/Tomita – Takesaki 整合
十分条件・反例・`kms_compat_test` と `kms_policy` の仕様。

付録 K: ノルム橋渡しベンチ（更新）
`op_probe` の**再試行/restarts**、**非収束時の扱い**（`INCONCLUSIVE` or `WARN`）を定義。

付録 L: MWE（最小動作例）
- **TG-Ind 検証**呼び出し例
- **パイプライン**実行例
- **ROC レポート**の JSON Lines 例

変更履歴（v2.4R1 → v2.4R2）
- TG-Ind 検証プロトコルを**規格化**（τ_Frob/τ_orth とログ仕様）。
- ホロノミー下界の**誤差予算**を**可計算化**（Richardson 残差＋線形係数＋校正法）。
- 外挿失敗検出の**ROC 校正**と**系別推奨値**を導入。
- **KMS 整合**に**反例＋フォールバック**（`kms_policy="geometric_only"`）。
- **API 型定義／エラー規約**を追加、**クイックスタート／MWE**を整備。
- **コスト・スケーリング報告**と**感度分析**を規格化。
- **op-probe** の**収束規約**（restarts/on_nonconverge）を明記。