CNNA – Gesamt-Plan für Refactoring und Strengthening des strikten Neuaufbaupfads

Knappfassung auf Basis des aktuellen Repos

Jan Seeck (antaris) als Editor, ChatGPT als Ghostwriter

Stand: 24. April 2026, fortgeschrieben nach gruener S9h-Schliessung, konsolidiertem gruenden A11-Arithmetikstand, revidierter A1–A6-Bewertung, eingefuehrtem Strengthening-Block A1b–A6b, re-anchored A-Planung auf der Spektral-Dualwurzel und aktualisierter Folgeplanung ab S10a

---

1 Leitsatz

Empfehlung: kein Totalneustart des Repos, aber ein **aktiver Pfad-Neuaufbau** im selben Repo. Der Refactor-Schritt 1 ist mit grünem P27 als Architektur-, Cutover- und Entkopplungsschritt abgeschlossen. Der nun folgende Schritt 2 ist das **Strengthening**: Aus dem bereinigten Pillar-A-Pfad soll ein tatsächlich rechnender, produktiv nutzbarer Generator werden, der nicht nur eine isolierte Smoke-Test-Instanz, sondern einen konkreten robusten Referenzfall des IDEAL-ToC und parallel dazu eine Familie legaler Substratinstanzen tragen kann. Der alte Pfad dient nur noch als **Regressionsorakel**; nach erfolgreichem modulweisem Cutover ist er aus dem aktiven Build-Pfad nach `legacy_sources/CNNA_v0.100_unstable` verschoben. Die zwischenzeitliche dünne Step-1-Gate-Stufe war nur Abschlussaggregator und ist nach grünem P27 selbst in die Archivspur überführt; die aktive Produktionslogik liegt nun vollständig in den fachlich zuständigen Pillar-A-Modulen. Strengthening bedeutet dabei ausdrücklich nicht, einen Smoke-Test-Seitenpfad einzuziehen, sondern dieselbe geared Hauptkette von der Wurzel bis zum Handoff so zu verschärfen, dass sie als später tragender Produktionspfad bestehen bleibt.

Prinzipien

- **Derived-only lokal:** jedes aktive Modul trägt sein starkes Artefakt selbst.
- **Wurzelprinzip:** derived-only beginnt beim fundamentalsten Objekt des aktiven Pfads und läuft strikt nach vorn.
- **Geared global:** jeder Nachfolger ist der kanonische Consumer des Vorgängers; keine freien Seiteneingänge und keine Bypässe.
- **Import-Vorabprüfung:** tragende Imports müssen selbst bereits derived-only, nicht-zirkulär und nicht vorausnehmend sein.
- **Legacy-Politik:** nur direkt oder fast direkt migrierbare Legacy-Dateien werden portiert; alles andere wird neu geschrieben.
- **Archivdisziplin:** nach erfolgreichem Cutover wird alter Parallelcode aus dem aktiven Build-Pfad in `legacy_sources/CNNA_v0.100_unstable` verschoben; er bleibt dort als Regressionsorakel erhalten, wird aber nicht dauerhaft als aktive Doppelspur mitgeführt.
- **Ideal-zuerst-Prinzip für das ToC-Substrat:** Das ToC wird im aktiven Pfad zuerst als ideales unendliches Substrat modelliert. Reale endliche Objekte sind davon strikt abgeleitete Approximanten und nicht umgekehrt.
- **Fixpunktprinzip für den Grenzpfad:** Die intendierte Aussage $\mathrm{REAL}_\infty \to \mathrm{IDEAL}_\infty$ ist nicht als rohe Gleichsetzung zweier Typen zu lesen, sondern als gerichtete Rekonstruktion des idealen ToC aus einer Approximantenfamilie. Diese Lesart umfasst ausdrücklich das Verschwinden aller

Rand-, Cutoff- und Umgebungsartefakte im idealen Fixpunkt sowie die Kontraktion aller endlichen Zellen ins Infinitesimale unter wachsendem Umgebungsdruck.

- **Dualpräsentationsprinzip für** IDEAL_∞**:** abstrakter Idealvertrag und adressierte Präsentation werden nicht roh identifiziert, sondern über eine strikt strukturtreue Äquivalenz mit beidseitigen Transfersätzen verbunden.
- **Kanonizitätsprinzip für die Box-Inhalte:** P4 schließt zunächst nur die kanonische Adressierung der idealen Zellseite. Die kanonische Etikettierung des gesamten "Box-Inhalts" – insbesondere Kanten, Schnitte, Approximanten, Rand, Komplement/Umgebung, IR-Anteil und UV-tail – darf erst als derived-only Folge der P5-invarianten Transportdaten und ihrer späteren Consumer entstehen.
- **Generatorprinzip für Schritt 2:** der produktive Smoke-Test darf kein Sonderpfad sein. Referenzfall und Substratvariationen müssen denselben Downstream-Pfad konsumieren, der später auch die Pillar-Handoffs trägt.
- **Familienprinzip für das Substrat:** neben einem robusten konkreten Referenzfall des IDEAL-ToC muss der aktive Pfad eine Familie legaler Substratinstanzen tragen; die Variation ist nur dann zulässig, wenn dieselbe Contract-/Addressing-/Equiv-Logik und dieselben späteren Consumer erhalten bleiben.
- **Keine freien Operatorzeugen im Produktionspfad:** freie Witnesses wie Gewichte, Inversen oder Regularisierungsshifts sind im Strengthening nicht bloß hübsch zu kapseln, sondern soweit möglich in legale Provenienzketten des aktiven Pfads zu internalisieren.
- **Inputvektorprinzip des Generators:** produktive Pillar-A-Läufe variieren nicht nur über das Substrat, sondern über den Dreiklang *(Substrat, WeightPolicy, Cutoff)*; universelle Struktursätze sind deshalb von genuinely variationsabhängigen Aussagen explizit zu trennen.
- **Computability-Budget-Prinzip:** für jedes Zahnrad des Root-to-Handoff-Generators ist explizit festzuhalten, ob es `computable`, nur intern lokal `noncomputable` oder nach aktuellem Stand noch offen ist; der globale Generator darf nicht schleichend durch viele scheinbar lokale Ausnahmen in einen diffusen `noncomputable`-Status kippen.
- **Parallel-Dualstrang-Prinzip für algebraische Seeds:** wo eine A-seitige endliche algebraische oder operatorische Oberfläche zugleich als ausführbare Generatorwurzel und als spätere analytische Consumer-Oberfläche benötigt wird, ist sie als expliziter Parallelstrang zu modellieren: ein operativer, öffentlich computabler Produktivstrang über einem ausführbaren Koeffiziententyp (im Strengthening zunächst `ExecComplex`) und ein davon getrennter, gegebenenfalls `noncomputable`er Spiegelstrang über **C**. Beide Stränge dürfen nur über eine bewiesene strukturtreue und injektive Brücke miteinander verbunden werden. Der operative Generatorpfad konsumiert ausschließlich den computablen Strang; der **C**-Strang dient Analyse, Transfer und späteren Pfeilern, darf aber nicht stillschweigend zum operativen Ausgang des strikten Pfads werden.
- **Norm-Dualstrang-Prinzip:** die Frobenius-Norm in `MatrixNorms` ist operative Vergleichs- und Shift-Norm des `ExecComplex`-Generators ohne ontischen Anspruch. Die ontische Normkandidatin fuer die Cayley-Dickson-Skalarstaffelung wird dagegen nicht aus `MatrixNorms` entnommen, sondern auf der geschlossenen S8-Spektralwurzel als basisinvariante, star-abgeleitete Spektralnorm der Form $\mathrm{tr}(aa^*) = \sum_i \lambda_i^2$ gelesen; diese kreuztermfreie Normlesart fliesst erst in S16–S19 als natuerlicher Kandidat der ontischen Skalarschichtung ein. Damit bleibt der primaere strukturelle Beweisort der normbezogenen Kreuzterm- und Kompositionsanalyse weiterhin S6b.
- **Prae-numerisches Primat der Sektorstruktur:** die helle/interface/dunkle Sektorzerlegung ist ontisch und im Plan begrifflich vor jeder Skalarwahl anzusetzen; Zahlen, Operatoren und Kopplungsalgebren dürfen ihre Rolle erklären, aber nicht die Existenz der Komplementstruktur

selbst begründen.

- **Ontisches Schichtungsprinzip fuer Skalaren:** der Plan unterscheidet fortan explizit zwischen prae-numerischer Struktur, einer reell-positiven Einzelsektor-Schicht $\mathbf{R}_{\geq 0}$, der komplexen bright-dark-Kopplungsschicht $\mathbf{C}$ sowie spaeteren Quaternionen-/Oktonionen-Erweiterungen $\mathbf{H}, \mathbf{O}$; diese Schichten duerfen weder mathematisch noch proseartig ineinander verschliffen werden.

- **Statusdisziplin fuer Scalar-Emergence-Aussagen:** maschinengepruefter A-Code, etablierte externe Saetze, strukturelle Beobachtungen, komparative Forschungslesarten, offene Forschungsfragen und Fernziele werden im Plan ausdruecklich getrennt gefuehrt; insbesondere darf „C aus Sektorkopplung“ nicht als bereits formal bewiesener Satz ausgegeben werden, solange nur eine proof-carrying Constraint- bzw. Hypothesenoberflaeche vorliegt.

- **Statusdisziplin fuer Spektral-Regime-Recovery ab gruener S8-Schliessung:** der Abschluss der endlichen Spektralwurzel bedeutet nur, dass die operative algebraische Spektraloberflaeche des Generators proof-carrying geschlossen ist. Externe harte Saetze wie Kesten 1959 fuer den unendlichen regulaeren Baum oder McKay 1981 fuer endliche lokal baumartige regulaere Graphfolgen, CNNA-interne Gap-/Masse-Kandidaten, Levelfolgen als moegliche RG-Observablen und spaetere Norm- oder Kompositionspflichten sind davon strikt zu trennen und duerfen nach gruener S8-Schliessung weder als bereits bewiesene A-Saetze noch als weitere Unterphasen S8g+ vermischt werden.

- **Code-Primat- und Dokumentationstrennungsprinzip:** repo-interne Primaerquelle ist ausschliesslich der maschinengepruefte Lean-Code samt gruener Toolchain. Aktive Lean-Module tragen deshalb nach Moeglichkeit keine inhaltstragenden Kommentare; erklaerende, motivische, architektonische oder vergleichende Prosa gehoert stattdessen in modulweise externe Dokumentation *(Markdown/PDF)*. Diese Dokumentation ist bewusst sekundar, interpretativ und potentiell fehlbar; sie darf den formal geprueften Code weder semantisch ersetzen noch in Statusfragen ueberstimmen.

- **Regime-Einbettungs- und Falsifikationsprinzip:** etablierte physikalische Theorien sollen im Programm – sofern CNNA traegt – als abgeleitete Spezialregime, Grenzfaelle, effektive Beschreibungen oder Quotienten der fundamentaleren bright/interface/dark-Struktur wiederauffindbar sein. Pillar A hat dafuer alle A-seitig derived-only vorbereitbaren algebraischen, operatorischen, zustandsraeumlichen, thermischen, kanalischen, limesseitigen und diagnostischen Seeds bereitzustellen, darf spaetere Fachlogik aber nicht als bereits geschlossen vorwegnehmen. Gerade dadurch bleibt das Programm zugleich integrativ und falsifizierbar: das Ausbleiben einer sauberen Regime-Rueckgewinnung ist nicht nur eine offene Baustelle, sondern ein echter Drucktest gegen die Tragfaehigkeit der Gesamtarchitektur.

- **Cayley-Dickson-Fortsetzungsprinzip:** jede weitere algebraische Verdopplung oberhalb der operativen `ExecComplex`-Schicht ist nur als explizit markierter Folgepfad zulaessig; weder $\mathbf{H}$ noch $\mathbf{O}$ duerfen den produktiven $\mathbf{C}$-Pfad stillschweigend ersetzen. Zulaessig bleiben dagegen spaetere explizit benannte Theorem- oder Consumerpfade, in denen quaternionische bzw. oktonionische Strukturen dark-sektor- oder randgebunden konsumiert werden, falls dies mathematisch wirklich benoetigt und sauber bewiesen ist; verboten ist allein der globale, implizite Koeffizientenwechsel.

- **A-interne Strukturpfade fuer Scalar-Emergence und Cayley-Dickson:** diese Bereiche gehoeren in Pillar A, aber nicht in den Handoff. Sie sind produktive A-interne Folgepfade des Generators, jedoch kein zweiter Root-Generator; sie duerfen nur pre-handoff A-Kernobjekte konsumieren und insbesondere weder `Outputs/A_to_*` noch `ABHandoffStrong` noch `PillarAStep1Closed` als internen Ersatzpfad verwenden.

- **Hurwitz-Stopprinzip:** der Satz, dass normierte Divisionsalgebren ueber $\mathbf{R}$ nur in den Dimensionen 1,2,4,8 auftreten, darf als externer harter Randstein des Plans konsumiert werden;

eine CNNA-interne Identifikation dieses Stopps mit dem Ende der Komplementhierarchie bleibt dagegen eigene Forschungsarbeit und ist als solche auszuweisen.

- **Instanzpropagationsprinzip für endliche Träger:** jedes Modul, das endliche Carrier, Teilmengen oder Prädikate über endlichen Typen exponiert, muss die nötigen `Fintype`-, `DecidableEq`- und `Decidable`-Instanzen explizit mitliefern oder theorematisch zurückgewinnen; vermeidbare `classical`-Abkürzungen sind im aktiven Pfad nicht zulässig.
- **A-seitige Vorstufenpflicht für spätere Pfeiler:** wenn spätere Pfeiler spektrale, thermische, zustandsräumliche, kanal- oder limesseitige Werkzeuge benötigen, die direkt aus A-generierten endlichen Operator- und Trägerdaten hervorgehen, dann sind diese als finite Seeds bereits in Pillar A bereitzustellen und nicht in B/C/D/E nachzubauen.
- **Algebraisches Wurzelobjektprinzip:** soweit spätere Pfeiler *-Algebra, Adjungierte/Hermitizität, endliche Hilbertraumstruktur, Kommutatorlogik, Zeitentwicklung oder Zustandsraum direkt aus endlichen A-seitigen Matrixcarriern und Operatoren gewinnen können, sind diese Strukturen bereits in A als fachlich getrennte Seeds zu liefern.
- **Legacy-Extraktionsprinzip jenseits von A:** aus REALOQS dürfen allgemeine mathematische Lemmata, Bright-Recovery-Vergleichsdaten, BInterface-Ideen und spätere Seed-Kandidaten extrahiert werden; unzulässig ist dagegen jeder Rückfall in boundary-first-Semantik, A→B-Zirkularität oder fachlich querliegende Sammelmodule.
- **Pfeilersemantik-Regel:** A liefert nur solche theoremisierte Axiomsamen, die aus A-generierten Größen derived-only erzeugt werden können; alles, was darüber hinaus echte AQFT-, OQS-, Geometrie- oder Matter-Fachlogik ist, wird in späteren Pfeilern aufgebaut und im Plan ausdrücklich als Folgearbeit getrennt geführt.
- **Fachmodulprinzip:** auch im Strengthening trägt jedes neue Modul eine zentrale Aussage bzw. Fachlogik. Höhere Modulanzahl ist zulässig, wenn dadurch Verzahnung, Wartbarkeit und Lesbarkeit steigen; verdeckte Sammelmodule sind zu vermeiden.
- **Gate-statt-Zeitplan-Prinzip:** es gibt bewusst keinen Kalender-, Deadline- oder Aufwandskalender. Angearbeitet wird stets das naechste fachlich anstehende Zahnrad; Fortschritt wird ueber grüne Builds, geschlossene Gates, explizit reduzierte offene Whitelist-Reste und geglueckte Referenz-/Variationslaeufe gemessen, nicht ueber Datumsversprechen.
- **Toolchain-Pinning-Prinzip:** der aktive Strengthening-Pfad arbeitet gegen eine gepinnte Lean-/Mathlib-Toolchain des Repos; fuer den aktuell gepruefften Stand ist dies Lean 4 `v4.28.0` mit `mathlib4` auf `v4.28.0` samt im `lake-manifest` fixierter Revisionslage. Upgrades duerfen nur in einer eigenen Kompatibilitaetsphase mit gruener Vollregression erfolgen und nicht still mitten im Abarbeitungspfad.
- **Editor-/Reviewer-Realitaetsprinzip:** der Plan ist so formuliert, dass er editorgefuehrt und single-thread robust abarbeitbar bleibt; AI ist Heuristik- und Ghostwriter-Schicht, spaetere fachliche Reviewer und moegliche weitere Contributor werden ausdruecklich erwartet, aber der Plan setzt ihre sofortige Verfuegbarkeit nicht voraus.
- **Audit-Durchsetzungsprinzip:** Regelkonformitaet ist nicht nur proseartig zu fordern, sondern lokal build- und grep-pruefbar zu halten: `lake build`, Importgraph, sowie scriptbare Audits auf `sorry`/`admit`/freie Axiome, `noncomputable`, `classical` und pauschale `@[simp]`-Flaechen gehoeren zu den Gates; CI ist nuetzlich, aber nicht Voraussetzung fuer die Geltung der Regel.

2 Projektvollzug, Referenzobjekt, Toolchain und Fallbacks

Warum es keinen Zeitplan gibt

Dieser Plan ist bewusst *kein* Kalenderplan. Die Roadmap selbst arbeitet pfeilerweise ueber Gates und partielle Ordnung statt ueber fixe Meilenstein-Daten; Strengthening folgt derselben Logik. Bearbeitet wird jeweils das fachlich naechste anstehende Modul bzw. die naechste Generatorengstelle. Ob ein Arbeitspaket Tage oder Monate braucht, ist nicht im Voraus zu versprechen; auf Kurs ist das Projekt dann, wenn der aktive Build grün bleibt, Gates sauber schliessen, Whitelist-Reste explizit schrumpfen und Referenz- sowie Variationslaeufe die jeweils behauptete Verzahnung wirklich tragen.

Arbeits- und Reviewer-Modell

Der aktuell realistische Vollzugsmodus ist editorgefuehrt: Jan Seeck (antaris) als Editor und Projektfuehrung, ChatGPT als heuristische Strukturierungs- und Ghostwriter-Schicht, spaetere fachliche Reviews fuer Mathematik, AQFT, OQS, Geometrie und Matter als ausdruecklich gewuenschte, aber nicht fuer jedes Zwischen-Gate vorausgesetzte Stufe. Der Plan ist deshalb absichtlich so geschrieben, dass er auch ohne grosses Parallelteam kohärent abarbeitbar bleibt; jede Phase soll fuer sich lokal prüfbar und ruecksetzbar sein.

Neu bindend ist dabei die Trennung zwischen formaler Quelle und Prosa: Im aktiven Pfad ist der maschinengepruefte Lean-Code die einzige repo-interne Wahrheitsquelle ueber den formalisierten Stand. Modulweise erklaerungen, Lesehilfen, Architekturkommentare, mathematische Einordnungen und Forschungslesarten werden deshalb bewusst aus dem Code ausgelagert und als externe Begleitdokumente pro Modul oder Modulgruppe gefuehrt, vorzugsweise als `.md`- oder `.pdf`-Artefakte. Diese Texte dienen Navigation, Review und Anschlussfaehigkeit, bleiben aber stets sekundar gegenueber dem gruenden Code und sind in Statusfragen ausdruecklich als interpretative, potentiell revidierbare Schicht zu lesen.

Gepinnte Toolchain statt gleitender Zielscheibe

Die aktuell gepruefte Repo-Lage ist toolchain-gepinnt: `lean-toolchain` fixiert Lean 4 `v4.28.0`, das `lakefile` bindet `mathlib4` an `v4.28.0`, und das `lake-manifest` fixiert die konkrete Revisionslage. Der Plan nennt diese Pinning-Strategie explizit, damit `mathlib`-Updates nicht unbemerkt Beweise oder Typklassenlagen verschieben. Jede kuenftige Versionserhoehung ist als eigene Kompatibilitaets- und Regressionsphase zu behandeln; waehrend einer laufenden S-Phase gilt die gepinnte Toolchain als normativer Zielzustand.

Was das konkrete IDEAL-ToC im Plan ist

Der Strengthening-Plan braucht einen *konkreten* robusten Referenzfall des IDEAL_∞, ohne daraus die gesamte Stammtheorie auf einen einzigen Substratstil zu verengen. Referenzobjekt ist deshalb der einfachste fachlich belastbare Fall eines **wurzelbasierten, unendlichen, regulär verzweigenden ToC** mit adressierbarer Zellfamilie; im praktischen Referenzlauf ist dies der reguläre b-aere rooted Tree-of-Cliques-Fall. Die zweite legale Substratfamilie bleibt keine unbekannte Blackbox, sondern ist im Plan bereits als **levelabhaengig verzweigender Baum** mit Branching-Funktion $\beta : \mathbf{N} \to \mathbf{N}$ vorgezeichnet. Weitere Substratfamilien sind zulaessig, aber erst dann architektonisch relevant, wenn sie denselben Contract-/Addressing-/Equiv-Pfad instanziieren.

Physikalischer Zieltyp des Programms

Das Projekt behauptet nicht, in Pillar A bereits eine volle emergente Raumzeit- oder AQFT-Theorie zu schliessen. Der belastbare A-seitige Zieltyp ist strenger und schmaler: A soll einen theoremisierten, berechenbaren und zielneutralen Generator liefern, aus dem spaetere algebraische, thermische, dynamische, kanalische, limesseitige und Diagnose-Samen derived-only hervorgehen. Zugleich ist das Programm nicht auf einen abstrakten Generator ohne Rueckbindung angelegt: falls CNNA traegt, muessen etablierte physikalische Theorien als Spezialregime, Grenzfaelle, effektive Beschreibungen

oder Quotienten der fundamentaleren Struktur wiederauffindbar sein. Pillar A bereitet dafuer alles vor, was A-seitig derived-only vorbereitbar ist; die eigentliche Regimepruefung und fachliche Rueckgewinnung kontinuierlicher Raumzeitsymmetrien, modularer Darstellungsarchitekturen, Typ-III-, CPT-, Kramers- oder Matter-/Gauge-Lesarten gehoert jedoch erst in B/C/D/E. Explorativ ist das Programm damit nur im Sinn spaeter physikalischer Emergenz; im Sinn der A-seitigen Generator- und Seed-Architektur ist es gerade *nicht* offen oder beliebig. Aus dem moeglichen Fundament soll auf diese Weise im weiteren Pfeilerzug ein belastbares Ja oder Nein werden.

Skalar-Emergenz und Statusgrenzen des Programms

Die Scalar-Emergence-Note erzwingt eine zweite, vom reinen Generatorabschluss verschiedene Planachse: Pillar A soll nicht nur eine ausfuehrbare algebraische Produktivwurzel bereitstellen, sondern auch die ontische Staffelung *prae-numerisch* $\to \mathbf{R}_{\geq 0} \to \mathbf{C} \to \mathbf{H} \to \mathbf{O}$ als explizit getrennte Statusschichten fuehren. Fuer den Plan bedeutet das: prae-numerische Sektorstruktur und operative `ExecComplex`-Wurzel gehoeren in den aktiven A-Pfad; die Aussage, dass **C** aus der Sektorkopplung erzwungen wird, ist zunaechst eine formal vorzubereitende Hypothese/Constraint-Oberflaeche; Quaternionen-, Oktonionen- und Drei-Generationen-Lesarten bleiben Folgearbeit mit eigenem Forschungsstatus. Zugleich fuehrt der Plan auf dieser Achse ausdruecklich komparative Forschungslesarten – etwa REALOQS als Einzelsektor-Ruecklese oder spaetere LQG-/Exzeptionalitaetsvergleiche – ohne sie mit maschinengeprueften Abschluessen zu verwechseln. Ein vollstaendiger formaler Beweis einer `ScalarClassification` ist damit ausdruecklich *nicht* Ziel des Generator-Kernblocks S0–S14, sondern ein Fernziel jenseits der reinen A-Produktivschliessung.

Neu bindend ist dabei die Architekturlesart der Folgephasen S15–S20: Sie bilden keinen top-level Begleitblock ausserhalb von A, sondern einen *A-internen produktiven Folgeblock* ueber dem bereits geschlossenen pre-handoff A-Kern. Dieser Block ist kein zweiter Root-Generator und darf den operativen **C**-Produktivpfad nicht ersetzen; er darf jedoch als legitimer theorematischer Consumer des A-Kerns laufen. `ComplexCouplingSeed` bzw. die S17-Lesart *ComplexEmergenceBridge* bleibt dabei zunaechst Constraint-/Hypothesenoberflaeche und kein Zwangssatz. `QuaternionicSeed` und `OctonionicSeed` sind entsprechend explizite Folge-Seeds bzw. Satzoberflaechen; sofern spaeter mathematisch wirklich benoetigt, duerfen **H**- bzw. **O**-gebundene Spezialpfade dark-sektor- oder randgebunden produktiv auftreten, aber nie als stiller Ersatz des globalen **C**-Strangs.

Gerade weil der Plan Zukunft entwirft und nicht nur Ist-Zustaende protokolliert, darf er dabei starke interne Leit-Hypothesen fuehren; nach aussen sind diese jedoch streng vom bereits formal Geschlossenen zu trennen. Dass beim Bauen neue, laenger werdende, aber lokalere und dadurch realistischere Hypothesenlisten herausfallen, ist in diesem Sinn kein Planversagen, sondern ein erwartbarer Gewinn an architektonischer Schaerfe.

Konvergenzbegriff fuer $\mathrm{REAL}_\infty \to \mathrm{IDEAL}_\infty$

Der Plan versteht diese Lesart zunaechst **algebraisch und gerichtet**, nicht als bereits abgeschlossene topologische Vollendung. `DirectedLimit` und spaetere `InfiniteCarrier-`/ `PreNet`-Seeds sollen stabile endliche Ausschnitte, Restriktions- und Uebergangsmorphismen sowie Eindeutigkeits- und Verträglichkeitsdaten formalisieren. Erst auf spaeteren Pfeilern wird daraus eine quasilokale, topologische oder operatoralgebraische Vervollstaendigung. Damit ist der Limesbegriff in A wohldefiniert genug fuer Export und Diagnose, ohne analytisch mehr zu behaupten als der aktuelle Pfad traegt.

Warum raw und stabilized getrennt bleiben

Der rohe Pfad ist im Plan nicht dekorativer Ballast. Er bleibt als Audit-, Diagnose-, Monotonie- und Regressionsflaeche sichtbar, damit die Stabilisierung nachpruefbar als kontrollierte Transformation derselben Provenienz erscheint und nicht als semantischer Austausch des Operators. Operativ konsumieren spaetere Zahnräder nur den stabilisierten Pfad; der rohe Pfad bleibt fuer Vergleich, Regularisierungsprovenienz, Kontraktionsexperimente und moegliche Fallback-Analysen erhalten.

Fallback- und Eskalationsregeln

- **Solver-Fallback:** Wenn `InteriorInverse` oder `interfaceInverse` in einer Phase nicht voll internalisierbar sind, darf als *einziger* Fallback ein lokaler interner Solver-/Reduktionsvertrag mit whitelist-faehigem, verborgenem `noncomputable`-Zeugen stehen bleiben. Unzulaessig bleibt dagegen, solche Inversen als freie oeffentliche Strong-Felder oder gar als spaeten Seiteninput zu behalten.
- **Mathlib-Fallback:** Falls `mathlib` benoetigte Spektral-, Matrixexponential- oder Algebra-Fakten nicht in brauchbarer Form liefert, werden zunaechst lokale CNNA-Hilfssaetze ueber der gepinnten Toolchain gebaut. Ein Fork oder Upstream-Patch ist erst letzte Eskalationsstufe, wenn die lokale Kapselung die Fachmodulgrenze nicht mehr sauber halten kann.
- **Variations-Fallback:** Falls eine weitergehende Substratfamilie noch nicht schliessbar ist, darf der Hauptpfad nicht blockiert werden: Referenzfamilie plus bereits legalisierte Variationsfamilie muessen weiter grün laufen; jede zusaetzliche Familie wird dann als separater nachgezogener Variationsblock behandelt.
- **Planstil-Fallback:** Die wiederkehrenden P0–P27-Schlussformeln im Bestandsblock sind bewusst Audit-Ledger und keine überfluessige Stilwiederholung. Wo alle Phasen dieselbe Restriktionslogik pruefen muessen, ist die nahezu formelhafte Wiederholung Teil der Nachvollziehbarkeit; verkuerzt werden darf nur, wenn dieselbe Gate-Pruefbarkeit erhalten bleibt.

3 Harte Repo-Befunde

- `CNNA/Gates/PillarAStep1Bridge.lean`: 1792 LOC, 40 `def`, 222 `theorem`, 28 `native_decide`, 1115 `concrete`-Treffer. Folge: die Brücke ist zu schwer.
- `CNNA/Core/Step1StrongCore.lean` enthält bereits echte Kopplung. Folge: eine brauchbare Wirbelsäule existiert.
- `CNNA/Core/ABHandoffStrong.lean` ist noch zu schmal. Folge: die A→B-Abgangswelle ist noch zu kurz.
- `CNNA/AQFT/*` ist überwiegend Stub-Hülle. Folge: Schritt 2 darf nicht auf der jetzigen AQFT-Schicht aufsetzen.
- Der Legacy-Bestand von REALOQS zeigt zugleich, dass `StarAlgebra`, `State`, `GNS`, `KMS`, `LocalNet`, `StateNet`, Kanal- und Lindblad-Pfade später tatsächlich benötigt werden, im Altbaum aber fachlich teils quer zu Derived-Modulen, `BInterfaces`, Bright-Recovery und Beispielen verteilt sind. Folge: Strengthening muss die A-seitig generierbaren algebraischen und handoffnahen Vorstufen in eigene Fachmodule zurückholen, statt sie später in B/C erneut kreuz und quer aufzubauen.
- `CNNA/Core/ToC/Concrete.lean` importiert `DirichletLaplacian`; `CNNA/OQS/DtN.lean` importiert `RegionNet`. Folge: der heutige Pfad ist schichtungsmäßig verschmiert und muss entkoppelt werden.
- Der neue Pfad `CNNA/PillarA/Foundation/SubstrateClass.lean` trägt bereits einen ersten idealen Substratkern mit Wurzel auf Ebene 0, Eltern-/Kinder-Kohärenz, Nichtterminalität, dynamischen Branching-Daten, `refine/coarsen`-Vorstufe und expliziten unendlichen Fäden; `CNNA/PillarA/Foundation/MatrixNorms.lean` ist kein Stub mehr. `MatrixNorms` ist damit der erste natürliche Anwendungsfall des algebraischen Dualstrangs: auf der operativen `ExecComplex`-Seite muss das Modul eine explizit computable Shift-, Vergleichs- und Positivitätsoberfläche (insbesondere Frobenius-Quadrat, Nulltest, Positivität bei $\neq 0$ und daraus konsumierbare Shift-Stufe) tragen, während analytische Frobenius-/Normfakten über **C** nur als getrennter Spiegelstrang sichtbar bleiben dürfen. Folge: P1 ist als Foundation-Vorstufe geschlossen, aber die leere-Ursprung-

/Fixpunktsemantik gehört noch präzise in P3; zugleich ist die bisherige MatrixNorms-Sonderregel in die allgemeine Dualstrang- und Whitelist-Logik zu überführen.

- `CNNA/Core/ToC/Contract.lean` importiert `ToC/Concrete`. Folge: P3 ist nicht als reiner ToC-Vertrag realisiert.
- Der aktuelle aktive ToC-Pfad modelliert noch kein ideales unendliches ToC, sondern ein endliches, ad-hoc parametriertes Ersatzobjekt. Folge: die Grenzlesart $\mathrm{REAL}_\infty \to \mathrm{IDEAL}_\infty$ ist ontisch noch nicht sauber aufgesetzt.
- Mit grünem P3 liegt nun ein erster formalisierter abstrakter Vertrag für IDEAL_∞ auf dem neuen Pfad vor. Folge: P0–P3 werden nicht zurückgenommen; stattdessen ist vor der vollen Approximantenfamilie explizit eine Adressierungs- und Äquivalenzschicht einzuplanen.
- Mit grünem P4 steht nun eine erste kanonische Adressierung der idealen Zellseite. Folge: diese Kanonizität darf noch nicht auf den gesamten "Box-Inhalt" extrapoliert werden; erst P5–P17 dürfen daraus schrittweise kanonische Identifikationen für Schnitte, Approximanten, Rand-/Komplementdaten sowie IR/UV-Anteile ableiten.
- Mit nun grünem P0–P8-Stand trägt der aktive neue Finite-Pfad bereits eine frühe Disziplin für `simp`-Politik, explizite Rekonstruktionssätze und entscheidbare adressierte Schnitte: die Zylinder-/Teilbaum-Schnittseite ist im aktiven Pfad computable, `RegionCore` und `BoundaryPorts` sind als getrennte Consumer-Stufen geschlossen, und der verbleibende nichtkonstruktive Rest sitzt nur noch in den unvermeidbaren Matrixnorm-Fakten. Folge: P9 darf auf einem bereits bereinigten konsumierenden Box-Pfad aufsetzen und soll diese Disziplin nicht wieder aufweichen.
- Mit nun grünem P0–P9-Stand ist `Finite/Approximant` als reine Consumer-Stufe von `BoundaryPorts` geschlossen: `ApproximantStrong` baut weder einen zweiten `Concrete`-Kern noch einen Seitenpfad zurück nach `RegionCore` auf, sondern fixiert den bereits kanonisch gewonnenen endlichen Box-Inhalt nur noch zum gewählten Cutoff. Die anfänglich zu breite `@[simp]`-Oberfläche wurde wieder zurückgenommen; im aktiven P9-Pfad verbleiben weder neue `noncomputable`-Restquellen noch klassische Hilfsabkürzungen. Folge: P10 bleibt rein bedingtes Supportmodul und wird nur aktiviert, wenn P11 es tatsächlich benötigt; andernfalls läuft der Hauptstrang direkt zu P11.
- Mit nun grünem P0–P10-Stand ist `Finite/Selection` als explizit isoliertes Supportmodul über `ApproximantStrong` geschlossen: `SelectionStrong` erzeugt keinen Alternativpfad, rekonstruiert weder `Concrete` noch `RegionCore`/`BoundaryPorts` neu und bleibt strikt Consumer des bereits kanonisch fixierten Box-Inhalts. Im aktiven P10-Pfad wurden keine neuen `noncomputable`-Restquellen, keine klassischen Hilfen und keine pauschalen `@[simp]`-Wrapper eingeführt; die modulnahe Notation bleibt parameter-sichtbar und rein lesbar. Folge: P11 kann `Selection` bei echter Teilstruktur-Auswahl konsumieren, muss dies aber architektonisch nicht; der Hauptstrang bleibt auf den Dirichlet-Kern fokussiert.
- Mit nun grünem P0–P11-Stand ist `Finite/DirichletLaplacian` als reiner Operator-Consumer über dem bereits kanonisch fixierten endlichen Box-Inhalt geschlossen: `DirichletLaplacianStrong` baut keinen neuen `Concrete`-/`Region`-Kern auf, benutzt `Selection` nicht als versteckte Basis und führt keine neuen klassischen Hilfen, pauschalen `@[simp]`-Flächen oder zusätzlichen `noncomputable`-Restquellen in den aktiven Pfad ein. Der operative Träger des Dirichlet-Kerns ist dabei nicht nur eine Topschicht, sondern ein gebundener endlicher Box-Träger über allen Levels $\leq$ Cutoff; dies fixiert zugleich die saubere Blockzerlegung für den nachfolgenden DtN-Kern. Folge: P12 kann nun direkt als binärer DtN-Consumer von `DirichletLaplacianStrong` aufsetzen, ohne Rückimport von `RegionNet` oder Rückweichung auf Alt-/Bridge-Pfade.
- Mit nun grünem P0–P12-Stand ist `DtN/DtN` als reiner binärer DtN-Consumer über `DirichletLaplacianStrong` auf dem neuen Pillar-A-Pfad geschlossen: `DtNStrong` konsumiert nur den bereits kanonisch fixierten Dirichlet-Kern, führt keine Rückimporte von `RegionNet`, keine neuen `@[simp]`-Flächen

und keine neuen `noncomputable`-Restquellen in den aktiven Pfad ein. Mit `InteriorInverse`, `boundaryOperator`, `interiorSolve` und den expliziten Abtriebssätzen zur Randfluss- und Innenblock-Gleichung steht damit die proof-carrying Oberfläche für P13 bereit. Folge: `DtNStabilized` darf nun nur noch auf diesem neuen P12-Pfad aufsetzen; der alte `CNNA/OQS/DtN`-Pfad bleibt bis zum Downstream-Cutover reines Regressionsorakel.

- Mit nun grünem P0–P13-Stand ist `DtN/DtNStabilized` als explizite Stabilisierungsstufe über dem neuen binären DtN-Pfad geschlossen: `DtNStabilizedStrong` trennt rohen Randoperator, symmetrisierte Hilfsstufe und primäre stabilisierte Operatoroberfläche explizit, fällt weder auf `CNNA/OQS/DtN.lean` zurück noch konsumiert es versehentlich den **C**-Spiegelstrang von `MatrixNorms`. Die operative Shift-Regularisierung bleibt explizit computable und stammt ausschließlich aus der `ExecComplex`-Seite von `MatrixNorms`; lesbare Notation darf diese Provenienz nicht verdecken. Folge: P14 kann auf einer semantisch schärferen Operatoroberfläche aufsetzen, ohne spätere Symmetrie-, Cutoff- oder Umgebungsdiagnostik bereits in P13 zu verwischen.

- Mit nun grünem P0–P14-Stand sind `Sectors/BranchPatch` und `Sectors/ComplementSectorFamily` als separate ontische Vorstufe über dem neuen P13-Pfad geschlossen: `BranchPatchStrong` konsumiert nur `DtNStabilizedStrong`, `ComplementSectorFamilyStrong` nur `BranchPatchStrong`; die kanonische äußere Umgebung/Komplementseite der Box wird über den äußeren Support explizit familienfähig gemacht, und die Datenoberfläche unterscheidet theorematisch zwischen root-zentrierten Ausschnitten ohne äußere Umgebung und fensterartigen Ausschnitten mit echter Umgebung/Komplementseite. Im aktiven P14-Pfad wurden keine neuen `noncomputable`-Produktionsdefinitionen, keine klassischen Hilfen und keine pauschalen `@[simp]`-Flächen eingeführt. Folge: P15 kann `SectorSplit` auf einer bereits kanonisch markierten Umgebungslage aufbauen, statt diese Unterscheidung erneut implizit zu rekonstruieren.

- Mit nun grünem P0–P15-Stand ist `Sectors/SectorSplit` als geometrisch-ontische Sektorzerlegung über der in P14 kanonisch gewonnenen Umgebungs-/Komplementseite geschlossen: `SectorSplitStrong` konsumiert nur `ComplementSectorFamilyStrong`, rekonstruiert die helle Seite strikt aus `source.patch` und baut keine konkurrierende Umgebungsdiagnostik auf. Die aktive Oberfläche stellt helle, Interface- und dunkle Carrier sowie die daraus derived-only rekonstruierte Interface-Region/-BoundaryPorts bereit; Präsenz bzw. Abwesenheit einer äußeren Umgebung ist theorematisch über die dunkle Seite unterscheidbar. Im aktiven P15-Pfad wurden keine neuen `noncomputable`-Produktionsdefinitionen, keine klassischen Hilfen und keine pauschalen `@[simp]`-Flächen eingeführt. Folge: P16 kann `BranchingWitness`/`SelectedBranching` direkt als proof-carrying Consumer von `SectorSplitStrong` formulieren und darf dabei weder die Umgebungsdiagnostik noch die Sektortrennung erneut aus Rohdaten rekonstruieren.

- Mit nun grünem P0–P16-Stand sind `Sectors/BranchingWitness` und `Sectors/SelectedBranching` als proof-carrying Witness-/Auswahlstufe direkt über `SectorSplitStrong` geschlossen: `BranchingWitnessStro` rekonstruiert aus der bereits kanonischen Sektorzerlegung eine explizite Menge zulässiger Branching-Levels samt Admissibility-Oberfläche; `SelectedBranchingStrong` konsumiert genau diese Witness-Stufe und schließt eine kanonische Minimalwahl, ohne Umgebungsdiagnostik, Sektortrennung oder Branching-Minimalität erneut aus Rohdaten aufzubauen. Im aktiven P16-Pfad wurden keine neuen `noncomputable`-Produktionsdefinitionen, keine klassischen Hilfen und keine pauschalen `@[simp]`-Flächen eingeführt; modulnahe Notation exponiert `BranchingWitness` und `SelectedBranching` parameter-sichtbar. Folge: P17 muss UV-/Branching-Selektion als Consumer dieser bereits kanonisierten Branching-Oberfläche formulieren und darf weder Aktivitätszähler noch rohen `SectorSplit`-Zugriff als Ersatz für die Witness-/Auswahlstufe verwenden.

- Mit nun grünem P0–P17-Stand sind `Sectors/UVSpectralSelector` und `Sectors/BranchingSelector` als deterministische/proof-carrying Selektorstufe über der bereits kanonisierten Witness-/Auswahloberfläche geschlossen: `UVSpectralSelectorStrong` markiert IR/UV-Anteile ausschließlich als derived-only Provenienzfläche über `SelectedBranchingStrong`, und `BranchingSelectorStrong` bündelt genau diese UV-Oberfläche mit der kanonischen Branching-Auswahl, ohne rohe Aktivitätszähler oder

direkten `SectorSplitStrong`-Zugriff als Ersatzpfad zu verwenden. Die im P17-Pfad auftretenden abhängigen Level-/Carrier-Beziehungen werden explizit theorematisch transportiert und nicht als definitorische Gleichheit kaschiert. Im aktiven P17-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die lesbare Notationsoberfläche für `UVSpectralSelector` und `BranchingSelector` ist freigegeben und importseitig eingebunden. Folge: P18 kann `ParameterClosure` nun als harten Consumer von `DtNStabilized` plus kanonisierter Branching-/UV-Selektoroberfläche formulieren.

- Mit nun grünem P0–P18-Stand ist `Closure/ParameterClosure` als harte Closure-Parameterstufe über `DtNStabilizedStrong` und `BranchingSelectorStrong` geschlossen: `ParameterClosureStrong` bündelt die bereits geschlossene Shift-/Stabilisierungsoberfläche des Operatorpfads mit der kanonisierten Branching-/UV-Seite, ohne einen Rückweg auf rohe `SectorSplit-` oder Witness-Daten als semantischen Ersatzpfad aufzubauen. Die aus P13 kommende computable Provenienz der Stabilisierung bleibt explizit sichtbar, und im aktiven P18-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die lesbare Notationsoberfläche für `ParameterClosure` ist lokal freigegeben und importseitig eingebunden. Folge: P19 kann `RegularizationClosure` nun direkt auf dieser Closure-Provenienz formulieren.

- Mit nun grünem P0–P19-Stand ist `Closure/RegularizationClosure` als eigentliche Regularisierungs-/Schließungsstufe über `ParameterClosureStrong` geschlossen: `RegularizationClosureStrong` konsumiert ausschließlich die bereits ausgewiesene computable Closure-/Shift-Provenienz, führt keine zweite nichtkonstruktive Regularisierungslogik ein und öffnet keinen Rückweg auf rohe `DtNStabilized-`, `BranchingSelector-` oder ältere OQS-Regularisierungspfade. Im aktiven P19-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die lesbare Notationsoberfläche für `RegularizationClosure` ist lokal freigegeben und importseitig eingebunden. Folge: P20 kann `RegionNet` weiter unabhängig als reinen Consumer von `SectorSplitStrong` formulieren, während spätere Closure-/Handoff-Consumer die abgeschlossene Regularisierungsseite nur noch über `RegularizationClosureStrong` und nicht rückwärts über `ParameterClosureStrong` oder ältere Altpfade konsumieren dürfen.

- Mit nun grünem P0–P20-Stand ist `Network/RegionNet` als reine Netzstufe über `SectorSplitStrong` geschlossen: `RegionNetStrong` konsumiert ausschließlich die bereits kanonische Sektorzerlegung, rekonstruiert weder Closure- noch Branching- oder ältere OQS/Bridge-Oberflächen als semantische Ersatzpfade und bündelt helle, Interface- und dunkle Regions-/Boundary-/Carrier-Daten nur als derived-only Projektionen derselben Netzquelle. Im aktiven P20-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die modulnahe Notationsoberfläche für `RegionNet` und `RegionKind` ist freigegeben und importseitig eingebunden. Folge: P21 kann `GeneralizedDtN` nun parallel zum Netzpfad als gekoppelten Consumer von `DtNStrong` und `SectorSplitStrong` formulieren, ohne `RegionNetStrong` oder die Closure-Seite als verdeckten Operatorersatz zu benutzen.

- Mit nun grünem P0–P21-Stand ist `Coupling/GeneralizedDtN` als gekoppelte Mehrsektor-DtN-Stufe parallel zum Netzpfad geschlossen: `GeneralizedDtNStrong` konsumiert ausschließlich `DtNStrong` und `SectorSplitStrong`, bindet deren Konsistenz explizit theorematisch und stellt sektorielle Regions-/Boundary-/Interior-/Carrier-Projektionen sowie blockweise Randoperator-Oberflächen als derived-only Kopplungsdaten bereit, ohne `RegionNetStrong`, `RegularizationClosureStrong` oder ältere Bridge-/OQS-Pfade als verdeckte Operatorersatzflächen einzublenden. Im aktiven P21-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die modulnahe Notationsoberfläche für `GeneralizedDtN` und `CoupledSectorKind` ist freigegeben und importseitig eingebunden. Folge: P22 kann `MultiSchur` nun als nächsten Coupling-Consumer formulieren; falls dafür lokal zusätzliche, fachlich zu `GeneralizedDtN` gehörige Strukturen in P21 nachträglich benötigt werden, dürfen diese dort ergänzt werden, sofern die Fachbereiche strikt getrennt in ihren jeweiligen Modulen verbleiben und `MultiSchur` keine semantische Fachlogik zurück in `GeneralizedDtN` verlagert.

- Mit nun grünem P0–P22-Stand ist `Coupling/MultiSchur` als Schur-Kompositionsstufe über der gekoppelten Mehrsektoroberfläche geschlossen: `MultiSchurStrong` konsumiert ausschließlich `GeneralizedDtNStrong`, trägt Restriktion, Gluing und die reduzierte Schur-Komposition explizit auf der aus P21 gewonnenen Kopplungsoberfläche und verlagert keine fachliche `MultiSchur`-Logik zurück in `GeneralizedDtN`. Rückwirkend in P21 ergänzte Hilfsstrukturen bleiben auf fachlich zu `GeneralizedDtN` gehörige Restriktions-/Projektionsdaten beschränkt; im aktiven P22-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt. Folge: P23 kann `InfiniteCarrier` nun als nächsten späten Network-Consumer formulieren, ohne die Coupling-Komposition erneut in Handoff- oder Netzmodule zu verschieben.

- Rauchtestbefund nach grünem P27: Die aktuelle Kopplungsseite trägt zwar den rohen binären `DtN`-Pfad und die Stabilisierung legal über `SectorSplitStrong`/`BranchPatchStrong` mit, die operative P21/P22-Oberfläche erzwingt diese Dualität aber noch nicht scharf genug. Folge: Die Architektur ist nicht durch einen Seitenpfad von `MultiSchur` nach `RegularizationClosureStrong` zu fixen; stattdessen ist `GeneralizedDtNStrong` selbst so fortzuschreiben, dass es raw/stabilized als explizit getrennte, theorematisch gebundene Operatoroberflächen derselben legalen P21-Provenienz trägt, während `MultiSchurStrong` ausschließlich den stabilisierten Restriktionspfad konsumiert und der rohe Pfad als Audit-/Diagnostikfläche sichtbar bleibt.

- Mit nun grünem P0–P23-Stand sind `Network/InfiniteCarrier` und `Network/SectorChannels` als späte Träger-/Kanalstufe über dem bereits geschlossenen A-Kern geschlossen: `InfiniteCarrierStrong` konsumiert ausschließlich die bereits geschlossene Netzstufe und bündelt keine neue ontische Unendlichkeitsquelle, sondern nur bereits vorhandene Foundation-/ToC-Unendlichkeitsdaten über eine explizite truncation-stabile Bindung an die aktive endliche Schicht; dadurch bleiben stabile endliche Ausschnitte und gerichtete Übergänge für spätere Rekonstruktions- und Limesanalysen exportierbar, ohne bereits AQFT-Quasilokalität oder Typ-III-Aussagen vorwegzunehmen. `SectorChannelsStrong` bleibt flankierender Supportconsumer der bereits geschlossenen Kopplungsoberfläche und rekonstruiert weder neue Netz-, Closure- noch Handoff-Struktur. Im aktiven P23-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die lesbare Notationsoberfläche für `InfiniteCarrier` und `SectorChannels` ist freigegeben und importseitig eingebunden. Folge: P24 kann `SectorSysEnv`, `RelativeEntropyFlow` sowie den Geometrieblock `LCPMeasure` → `Foliation` → `SpacetimePaths` als flankierende Supportschicht formulieren, ohne den Hauptpfad rückwärts zu beeinflussen.

- Mit nun grünem P0–P24-Stand sind `Network/SectorSysEnv`, `Network/RelativeEntropyFlow` sowie der Geometrieblock `Geometry/LCPMeasure`, `Geometry/Foliation` und `Geometry/SpacetimePaths` als flankierende Supportschicht über dem bereits geschlossenen A-Kern geschlossen: `SectorSysEnvStrong` konsumiert ausschließlich `SectorChannelsStrong` und führt die System/Interface/Environment-Seite nur als derived-only Reindexierung der bereits geschlossenen Netz-/Kanaloberfläche; `RelativeEntropyFlow` konsumiert ausschließlich `SectorSysEnvStrong` und bleibt entropische Flussbuchhaltung ohne neue physikalische Grundaxiomatik. Im Geometrieblock liefert `LCPMeasure` adressnahe Präfix-/Nähedaten, `Foliation` fixiert darüber nur die kanonische Blätterung nach Level, und erst `SpacetimePaths` konsumiert diese Blätterung für Weltlinien-, Tube- und Kausalbegriffe; Σ_k bleibt dabei Carrier-Schicht und nicht Slice-Ersatz. Im aktiven P24-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die lesbare Notationsoberfläche für `SectorSysEnv`, `RelativeEntropyFlow`, `LCPMeasure`, `Foliation` und `SpacetimePaths` ist freigegeben und importseitig eingebunden. Folge: P25 kann `SectorExport` nun als Exportconsumer der reifen A-Daten formulieren, ohne flankierende Netzwerk- oder Geometriebegriffe rückwärts in den Kernpfad zu verschieben.

- Mit nun grünem P0–P25-Stand sind `Handoff/SectorExport` und `Handoff/Step1StrongCore` als erste Handoff-Stufe über dem bereits geschlossenen A-Kern geschlossen: `SectorExportStrong` projiziert die reifen A-Daten aus dem bereits geschlossenen Netz-, Closure- und Coupling-Pfad

auf eine zielneutrale Exportoberfläche, ohne zur Sammelstelle roher Einzelinputs zu werden oder flankierende Netzwerk- bzw. Geometriebegriffe rückwärts in den Kernpfad zu verschieben. Die Provenienz der exportierten Daten bleibt explizit sichtbar; insbesondere wird die Regularisierungsseite nur über `RegularizationClosureStrong` konsumiert und nicht semantisch rückwärts über `ParameterClosureStrong` oder ältere Regularisierungspfade ersetzt. `Step1StrongCore` fasst darauf aufbauend nur bereits geschlossene Teilstränge kompakt zusammen, ohne selbst neue Fachlogik, neue Rohdatenpfade oder einen verdeckten Vorgriff auf den gerichteten A→B-Handoff einzuführen. Im aktiven P25-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt; die lesbare Notationsoberfläche für `SectorExport` und `Step1StrongCore` ist freigegeben und importseitig eingebunden. Folge: P26 kann nun `Handoff/Step1MathData`, `Handoff/ABHandoffStrong` und `Handoff/PillarAStep1Closed` als proof-carrying Abschluss der A-Abgangswelle formulieren; dabei bleibt `SectorExportStrong` die zielneutrale öffentliche Exportfläche von A, während `ABHandoffStrong` nur der gerichtete Adapter `A_to_B` ist und die zwischenzeitliche dünne Step-1-Gate-Stufe ausschließlich von `PillarAStep1Closed` gespeist werden darf.

- Mit nun grünem P0–P26-Stand sind `Handoff/Step1MathData`, `Handoff/ABHandoffStrong` und `Handoff/PillarAStep1Closed` sowie die zwischenzeitliche dünne Step-1-Gate-Stufe als proof-carrying Abschluss der A-Abgangswelle über dem bereits geschlossenen A-Kern geschlossen: `Step1MathDataStrong` bündelt das volle A-Mathematikpaket nur über `Step1StrongCore`, `ABHandoffStrong` bleibt ausschließlich der gerichtete Adapter `A_to_B`, `PillarAStep1Closed` ist das einzig legitime Endobjekt des geared Pfads, und die dünne Gate-Stufe konsumiert ausschließlich dieses Abschlussobjekt. Im aktiven P26-Pfad wurden keine neuen `@[simp]`-Flächen, keine neuen `noncomputable`-Produktionsdefinitionen und keine klassischen Hilfen eingeführt. Die globale Aliasoberfläche in `PillarA/Notation` für `Step1MathData`, `ABHandoffStrong` und `PillarAStep1Closed` ist freigegeben; die modulnahe Spiegelung in `Handoff/Notation` sowie ihre konsequente Konsumption wurde im Zuge des P27-Cutovers nachgezogen. Folge: der Altpfad-Cutover konnte abgeschlossen werden, indem alte Bridge-, Core-, OQS- und AQFT-Pfade aus dem aktiven Build-Pfad nach `legacy_sources/CNNA_v0.100_unstable` verschoben wurden; dies ist eine Archivierung außerhalb des Build-Pfads und keine fachlich aktive Doppelspur.

- Mit nun grünem P27-Stand ist der Archiv-Cutover abgeschlossen: Insbesondere `DirichletLaplacianBridge`, `StrongLegacyAdapters`, `RegularizationClosure_old` sowie weitere alte Core-/OQS-/AQFT-/Integration-/Bridge-Pfade und außerdem `Gates`, `Meta` und `Seeds` wurden nach `legacy_sources/CNNA_v0.100_` verschoben. Sie bleiben dort als Regressionsorakel erhalten, dürfen aber nicht mehr in den aktiven Build oder als semantisch gleichwertige Ersatzpfade zurückwirken.

- Der aktive Root-Build ist damit bewusst minimal: `CNNA/BuildAll.lean` importiert nur noch `CNNA/Notation` und `CNNA/PillarA/BuildAll`, während `CNNA/PillarB` bis `CNNA/PillarE` im aktiven Baum nur noch `BuildAll.lean`-Stubs tragen. Aktiv enthalten ist damit nur noch der neue derived-only Pillar-A-Pfad ausgehend vom ToC.

- Der aktive P27-Baum ist damit zwar architektonisch bereinigt und fast durchgängig generisch über `Cell` und `[SubstrateClass Cell]` parametrisiert, besitzt aber noch keine explizite Referenzfamilie eines konkreten IDEAL-ToC, die zusammen mit `Addressing` und `IdealAddressEquiv` als tatsächlich konsumierte Wurzelinstanziierung im aktiven Pfad läuft. Folge: der geared Pfad ist als Architektur geschlossen, aber noch kein produktiv nutzbarer Generator.

- Im nun gruenden Stand nach S7a trägt der Operatorpfad weiterhin freie bzw. weiter zu internalisierende Kernzeugnisse wie `weight` in `Finite/DirichletLaplacian`, `InteriorInverse` in `DtN/DtN` und `epsilon` in `DtN/DtNStabilized`; `interfaceInverse` tritt auf der Coupling-Seite dagegen nicht mehr als nackter Wurzelinput auf, sondern nur noch als innerhalb von P22 gebundene Interface-Eliminations- bzw. Reduktionslogik. Folge: Schritt 2 bleibt auf die weitere Legalisierung der verbleibenden Witnesses fokussiert; die P22-Schur-Seite ist diesbezueglich nicht mehr der primaere offene Wurzeleingang.

- Die Roadmap fordert zugleich, dass das ToC nicht auf ein einziges privilegiertes Substrat festgelegt wird und dass Closure-Parameter wie `b`, `Lmax`, β und sekundäre Regularisierer nicht als primitive Restparameter im Kern verbleiben. Folge: das Strengthening benötigt neben einer robusten Referenzfamilie auch eine explizite Variations- und Vergleichsoberfläche über gültige Substratfamilien.

- Im aktuellen gruenden S9-Grundgeruest sind `Finite/StateSpace`, `Finite/MatrixExponential`, `Finite/GibbsStateSeed`, `Finite/UnitaryEvolution`, `Finite/DynamicsAdapter` und `Finite/ChannelSeed` zwar build-stabil in `Finite/BuildAll`, `Finite/Notation` und `PillarA/Notation` eingezogen, planlogisch aber noch nicht geschlossen: ueber die sechs Dateien liegen derzeit 28 oeffentliche `noncomputable def`, `MatrixExponential` exponiert noch direkt `NormedSpace.exp`, `GibbsStateSeed` haengt oeffentlich an `mirror.eigenvalues` und `mirror.eigenvectorUnitaryMatrix`, `UnitaryEvolution` fuehrt Heisenberg- und Schroedinger-Konjugation im aktuellen Code noch formelgleich, und `ChannelSeed` bleibt bislang eine duenne algebraische Package-Schale ohne explizite Kraus-/Choi-/CPTP-Basis. Folge: S9 wird ab jetzt strikt in S9a–S9h als Haertungs- und Abschlussblock aufgefaechert; das gruende Grundgeruest ist Audit-Basis, aber noch kein zulaessiger Endzustand.

Neue Konsequenz aus der Scalar-Emergence-Note

Die neue Note aendert den Plan nicht nur proseartig, sondern architektonisch. Erstens verschiebt sie `ExecComplex` von einer bloss technischen Computability-Loesung zu einer expliziten ontischen Kopplungsschicht: Der operative **C**-Pfad ist weiterhin der einzig legale Produktivpfad, aber seine Rolle ist nun begrifflich an die bright-dark-Kopplung statt nur an mathlib-Umgehung gebunden. Zweitens verlangt sie eine harte Statusdisziplin: maschinengepruefte A-Strukturen, externe Saetze wie Hurwitz, strukturelle Beobachtungen, offene Forschungsfragen und Fernziele duerfen im Plan nicht mehr in einem einzigen Fortschrittsnarrativ vermischt werden.

Kritisch ist dabei festzuhalten, dass der aktuelle gruene P0–P27-Pfad die prae-numerische Prioritaet der Sektorstruktur bislang nur *lokal* und modulsemantisch, aber noch nicht als globalen Strengthening-Abschluss traegt: `SectorSplit` selbst importiert keine Skalarstruktur, der aktive Gesamtpfad erreicht die Sektorseite jedoch weiterhin ueber operatorische Vorgaenger. Genau diese Luecke darf der Plan nicht durch starke Formulierungen ueberspielen. Deshalb wird oberhalb des Generator-Kernblocks S0–S14 ein A-interner Scalar-Emergence-Block S15–S20 eingefuehrt, der lokale Prae-Numerik, explizite Einzelsektor-/Kopplungsseeds sowie Quaternionen-/Oktonionen-Folgepfade sauber trennt, ohne den produktiven **C**-Generatorpfad zu ersetzen. Produktiv ist dieser Folgeblock nur als theorematischer Consumer des bereits geschlossenen pre-handoff A-Kerns, nicht als zweiter Root-Generator und nicht als stiller Ersatz der operativen `ExecComplex`-Linie.

4 Vollständigkeitsprüfung: aktiver Pillar-A-Kern und A-interne Folgeblöcke

Fundamente: `SubstrateClass`, `ExecComplex`, `ExecComplexLemmas` (optional), `MatrixNorms`, `Interfaces`, `Determinism`, `HermitianStructure`, `FiniteHilbert`, `MatrixAlgebra`.
ToC-Idealkern und Präsentationsbrücke: `ToC/Contract`, `ToC/Addressing`, `ToC/IdealAddressEquiv`, `ToC/ConcreteIdeal`, `ToC/Concrete`.
Endliche Vorstufe: `Finite/CutSpec`, `RegionCore`, `BoundaryPorts`, `Approximant`, `Selection`, `DirichletLaplacian`, `SpectralDecomposition`, `SpectralDecompositionC`, `SpectralBridge` sowie als sichtbarer Folgeblock `Finite/ExecSpectral/*`; daran anschliessend das gruende, aber noch nicht planlogisch geschlossene S9-Grundgeruest aus `Finite/StateSpace`, `Finite/MatrixExponential`, `Finite/GibbsStateSeed`, `Finite/UnitaryEvolution`, `Finite/DynamicsAdapter` und `Finite/ChannelSeed`.
Sektor-/Branching-Vorstufe: `BranchPatch`, `ComplementSectorFamily`, `SectorSplit`, `BranchingWitness`, `SelectedBranching`, `UVSpectralSelector`, `BranchingSelector`.
OQS-Kern / Closure: `DtN`, `DtNStabilized`, `ParameterClosure`, `RegularizationClosure`, `GeneralizedDtN`, `MultiSchur`.
Netz / Geometrie / Handoff in Pillar A: `RegionNet`, `InfiniteCarrier`, `SectorChannels`,

`SectorSysEnv`, `RelativeEntropyFlow`, `Geometry/LCPMeasure`, `Geometry/Foliation`, `Geometry/SpacetimePath`
`SectorExport`, `Step1StrongCore`, `Step1MathData`, `ABHandoffStrong`, `PillarAStep1Closed`.
A-interner Scalar-Emergence-Folgeblock: `PillarA/ScalarEmergence/PreNumericSectorCore`,
`PillarA/ScalarEmergence/StatusLedger`, `PillarA/ScalarEmergence/OneSectorRealSeed`, `PillarA/ScalarE`
`PillarA/ScalarEmergence/QuaternionicSeed`, `PillarA/ScalarEmergence/OctonionicSeed`, `PillarA/Scalar`
A-interner struktureller CD-Pfad / Beweispflicht I: `PillarA/Structural/CayleyDickson/*`
als eigener struktureller Meta-Beweisstrang innerhalb von Pillar A, aber außerhalb von `PillarA/Handoff`;
Handoff-seitig erscheint davon höchstens eine dünne Abschlussoberfläche in `ProofObligationI/*`.
Alt-/Übergangsmodule: `DirichletLaplacianBridge`, `StrongLegacyAdapters`, `RegularizationClosure_old`

5 Pillar-Ordnungsregel für Refactor und Strengthening

- Neue A-Kernmodule gehen nach `CNNA/PillarA/...`; spätere neue B–E-Module würden nach `CNNA/PillarB/...` bis `CNNA/PillarE/...` gehen. Im aktuellen grünen P27-Stand bleiben `CNNA/PillarB` bis `CNNA/PillarE` im aktiven Baum jedoch bewusst auf `BuildAll.lean`-Stubs reduziert.
- Die Altordner `Core/OQS/AQFT/...` sowie weitere ersetzte Altpfade bleiben nur als Cutover-Quelle stehen und werden nach erfolgreicher Umstellung aus dem aktiven Build-Pfad nach `legacy_sources/CNNA_v0.100_unstable` verschoben; im grünen P27-Stand gilt dies zusätzlich für `Integration`, `Gates`, `Meta` und `Seeds`.
- Ab Start des Refactors wird keine neue Fachlogik mehr in alte Top-Level-Ordner geschrieben, sofern bereits ein neuer Pillarpfad existiert.
- Keine leeren Scaffold-Landschaften vorab; neue Unterordner und Module werden erst dann angelegt, wenn ihre Phase beginnt.
- Im ToC-Pfad gilt strikt: `Contract` modelliert den abstrakten idealen unendlichen Träger, `Addressing` seine adressierte Präsentation, `IdealAddressEquiv` deren strukturtreue Äquivalenz, `ConcreteIdeal` schließt den konkreten IDEAL-Referenzfall als rückgebundene Familie, und `Concrete` formuliert erst danach reale Approximantenfamilien dieses Trägers. Endliche Proxies dürfen weder den Idealvertrag noch die Adressierungs-Äquivalenz ersetzen; ebenso darf der konkrete Referenzfall nicht als stillschweigender Ersatz für den späteren realen Downstream-Pfad missverstanden werden.
- Kanonische Beschriftungen werden entlang dieses Pfads stufenweise geschlossen: P4 liefert die kanonische Zelladressierung des aktiven Ideals; P5 macht diese Präsentation transportinvariant; erst die späteren Consumer von P6 bis mindestens P17 dürfen daraus kanonische Identifikationen für Schnitte, Approximanten, Rand-/Komplementdaten, Environment-Splits sowie IR/UV-Anteile ableiten.
- Im Strengthening-Schritt dürfen neue A-Kernmodule nur dann ergänzt werden, wenn sie eine reale Generatorlücke schließen: computable Referenzfamilie, zweite legale Substratfamilie, explizite WeightPolicy-Achse, kanonischer Operatorseed, kontrollierte Solver-/Regularisierungsoberfläche, Generator- bzw. Variationsauswertung. Neue Module sind nicht als zweiter semantischer Pfad neben dem aktiven Baum zulässig; der einzige zulässige explizite Parallelfall ist der auditierbare algebraische Dualstrang aus computablem `ExecComplex`-Produktivpfad und getrenntem **C**-Spiegelstrang, dessen operative Konsumrichtung strikt festliegt.
- Der A-interne Scalar-Emergence-Block ist im Strengthening ein zusaetzlicher produktiver Folgepfad ueber dem bereits geschlossenen pre-handoff A-Kern: Er darf Statusledger, Prae-Numerik-Extraktion, komparative Forschungslesarten sowie Quaternionen-/Oktonionen-Seeds anlegen, jedoch weder einen zweiten Root-Generator neben dem `ExecComplex`-Pfad noch eine ueberschiessende Schlussbehauptung erzeugen. Alles, was dort ueber die operative **C**-Schicht hinausgeht, ist im Plan ausdruecklich als Seed, Hypothese, komparative Lesart oder Fernziel zu markieren; explizite **H**- oder **O**-Spezialpfade bleiben nur opt-in und dark-sektor- bzw. randgebunden zulaessig.

- Referenzfall und Substratfamilie dürfen insbesondere keinen parallelen Downstream-Strang erzeugen: ab dem ersten legalen `ToCStrong`- bzw. Approximantenkern müssen beide dieselben späteren Consumer bedienen. Smoke- und Analyseläufe gelten nur dann als architektonisch zulässig, wenn sie genau diesen Produktionspfad benutzen.

- Notationsdisziplin ab S3/S4: neu freigegebene Alias- und Referenznotationen gelten erst dann als wirklich geschlossen, wenn sie nicht nur in `ToC/Notation` bzw. `PillarA/Notation` definiert und importiert, sondern in den realen Consumern des gemeinsamen Generatorpfads sichtbar konsumiert werden. Bloß globale Spiegelung ohne operative Konsumption ist nach S3 nur als kurzer Zwischenstand zulässig und muss spätestens in S4 nachgezogen werden.

- Dokumentationsdisziplin des aktiven Pfads: inhaltstragende Kommentare sind aus aktiven Lean-Modulen nach Moeglichkeit zu entfernen statt als zweite semantische Spur im Code mitzuwachsen. Zweck, Exportrolle, mathematische Lesart, Handoff-Bedeutung, externe Literaturanker und offene Interpretationspunkte werden modulweise ausserhalb des Codes dokumentiert; zulaessige Primaerquelle fuer Repo-Aussagen bleibt jedoch immer der gepruefte Code selbst.

Zielstruktur für Pillar A sowie A-interne Folgebereiche

- `Foundation/`: `SubstrateClass`, `ExecComplex`, `ExecComplexLemmas` (optional), `MatrixNorms`, `Interfaces`, `Determinism`, `HermitianStructure`, `FiniteHilbert`, `MatrixAlgebra`
  mit Fokus auf idealen geschichteten Träger, Wurzelstruktur auf Ebene 0, lokaler Endlichkeit, Eltern-/Kinder-Kohärenz, nichtterminaler Verzweigung, dynamischen Branching-Daten, `refine/coarsen`-Vorstufe, expliziten unendlichen Fäden sowie einer nun explizit öffentlich computablen algebraischen A-Wurzel über `ExecComplex`; `MatrixNorms` bleibt der erste Dualstrang-Consumer dieser Wurzel, und die präzise leere-Ursprung-Semantik wird daran anschließend in `ToC/Contract` fixiert

- `ToC/`: `Contract`, `Addressing`, `IdealAddressEquiv`, `ConcreteIdeal`, `Concrete`
  wobei `Contract` den abstrakten idealen ToC-Fixpunkt trägt, `Addressing` zunächst nur die kanonische Zelladressierung des aktiven Ideals aufbaut, `IdealAddressEquiv` diese Präsentation strikt strukturtreu und transportinvariant zum abstrakten Vertrag zurückbindet, `ConcreteIdeal` darauf den konkreten IDEAL-Referenzfall als rückgebundene Familie schließt, und `Concrete` erst danach reale Approximantenfamilien sowie weitere kanonische Identifikationen als Consumer derselben Äquivalenz formuliert; die in S3 freigegebene Referenznotation ist dabei ab S4 im realen Downstream aktiv zu konsumieren und nicht nur global zu spiegeln

- `Finite/`: `CutSpec`, `RegionCore`, `BoundaryPorts`, `Approximant`, `Selection`, `DirichletLaplacian`

- `DtN/`: `DtN`, `DtNStabilized`

- `Sectors/`: `BranchPatch`, `ComplementSectorFamily`, `SectorSplit`, `BranchingWitness`, `SelectedBranching`, `UVSpectralSelector`, `BranchingSelector`

- `Closure/`: `ParameterClosure`, `RegularizationClosure`

- `Network/`: `RegionNet`, `InfiniteCarrier`, `SectorChannels`, `SectorSysEnv`, `RelativeEntropyFlow`

- `Geometry/`: `LCPMeasure`, `Foliation`, `SpacetimePaths`

- `Diagnostics/`: `SpectralDimension`, `SymmetryCutoffLimit`, `ScaleBreaking`, `ApproxSymmetry`, `DualObserverSeed`, `InterfaceEntropyBound`, `SpectralGapSeed`, `SchurBlockSymmetrySeed`, `HorizonDefini` `CayleyDicksonSpectralShift`; dazu `VariationAnalysis`, `DirectedLimit` sowie `Update/MismatchSeed`, `Update/TailEliminationSeed`, `Update/UpdateStep` als diagnostische und regimevorbereitende Supportschicht unterhalb spaeterer Pfeiler

- `Coupling/`: `GeneralizedDtN`, `MultiSchur`

- `Structural/`: `CayleyDickson/*` als eigener struktureller Meta-Beweisbereich fuer den CD-Pfad und Beweispflicht I; dieser Bereich liegt innerhalb von Pillar A, aber ausserhalb der gerichteten Handoff-Schicht
- `ScalarEmergence/`: `PreNumericSectorCore`, `StatusLedger`, `OneSectorRealSeed`, `ComplexCouplingSeed`, `QuaternionicSeed`, `OctonionicSeed`, `HurwitzStopSeed`
  als A-interner produktiver Folgebereich ueber dem generatorisch bereits geschlossenen pre-handoff A-Pfad; dieser Block ist kein normaler Pillar und kein zweiter Root-Generator. ***`ComplexCouplingSeed`*** *beschreibt die operative* **C***-Schicht nur als formal vorbereitete Kopplungsconstraint-Oberflaeche, waehrend Quaternionen-/Oktonionenmodule explizite Folge-Seeds bzw. spaetere opt-in Satzoberflaechen bleiben; der Cayley-Dickson-Pfad wird hier als legaler upstream-Seed aus* ***`PillarA/Structural/*`*** *konsumiert*
- `Handoff/`: `Core/*`, `Outputs/A_to_B`, `Outputs/A_to_C`, `Outputs/A_to_D`, `Outputs/A_to_E`, `Inputs/B_to_A`, `Inputs/C_to_A`, `Inputs/D_to_A`, `Inputs/E_to_A`, `ProofObligationI/*`, `ProofObligationII/*`, `ProofObligationIII/*`; bestehende Module wie `SectorExport`, `Step1StrongCore`, `Step1MathData`, `ABHandoffStrong` und `PillarAStep1Closed` werden in diese Semantik einsortiert statt als flache Sammlung weitergefuehrt; `ProofObligationI/*` bleibt dabei ausdruecklich nur duerre Ledger- bzw. Exportoberflaeche fuer die A-internen Strukturpfade
- `Aktiver Root-Build nach P27/S20: CNNA/BuildAll` → `CNNA/Notation` → `CNNA/PillarA/BuildAll`; `Gates`, `Meta` und `Seeds` liegen nur noch in der Archivspur, `PillarB`–`PillarE` bleiben im aktiven Baum auf `BuildAll.lean`-Stubs reduziert. Innerhalb von `PillarA/BuildAll` werden spaetestens ab S20 auch `PillarA/Structural/BuildAll` und `PillarA/ScalarEmergence/BuildAll` konsumiert; separate top-level Begleitbloecke `CNNA/Structural/*` oder `CNNA/ScalarEmergence/*` gehoeren dann nicht mehr zur Zielarchitektur

6 Legacy-Politik

6.1 Direkt oder fast direkt migrierbar

- `AQFT/StarAlgebra.lean`, `AQFT/CStarNorm.lean`, `AQFT/State.lean`, `AQFT/RelativeEntropy.lean`
- `AQFT/LocalNet.lean`, `AQFT/TimeReversal.lean`
- `AQFT/QuasiLocal/Carrier.lean`, `Structure.lean`, `StarAlgebra.lean`, `Completion.lean`

6.2 Nur nach Adapter-Stufe migrierbar

- `AQFT/StateNet.lean`
- `AQFT/GNS.lean`, `AQFT/KMS.lean`
- `AQFT/QuasiLocal/StateLift.lean`

6.3 Nicht migrieren, nur als Vorlage/Orakel verwenden

- `AQFT/Derived/BoundaryMatrixLocalNet.lean`
- `AQFT/Derived/BoundaryMatrixGNS.lean`
- `AQFT/Derived/BoundaryMatrixSplitProperty.lean`
- `AQFT/Derived/BoundaryMatrixQuasiLocalClosure.lean`
- `AQFT/TimeOrientation.lean`

6.4 Für P1–P6 besonders relevante Legacy-Anker

- `legacy_sources/REALOQS/PillarA/Core/SubstrateClass.lean`
- `legacy_sources/REALOQS/PillarA/Core/MatrixNorms.lean`
- `legacy_sources/REALOQS/PillarA/Core/Determinism.lean`
- `legacy_sources/REALOQS/PillarA/Ideal/Substrate.lean`
- `legacy_sources/REALOQS/PillarA/Ideal/Adapter/SubstrateInstance.lean`
- `legacy_sources/REALOQS/PillarA/Ideal/TreeOfCliques/Vertex.lean`, `Boundary.lean`, `CoarsenBoundar`
- `legacy_sources/REALOQS/PillarA/Ideal/Foliation.lean`
- `legacy_sources/REALOQS/PillarA/Ideal/LCPMeasure.lean`
- `legacy_sources/REALOQS/PillarA/Ideal/SpacetimePaths.lean`
- `legacy_sources/REALOQS/PillarA/Core/InfiniteCarrier.lean`

Diese Dateien sind keine Blaupause zum unkritischen Portieren. Sie markieren aber den mathematisch relevanten Suchraum: adressbasierte unendliche Schichtung, Ebenenmengen, Kinder-/Randabbildungen, Koarsenierung, Präfix-/Frontier-Ideen, eine kanonische Blätterung nach Level sowie ein expliziter Begriff eines unendlichen Trägers. Für `Addressing`, `IdealAddressEquiv` und spätere `CutSpec`-/Geometriemodule sind sie Inspirationsquelle, nicht automatische Pfadvorgabe. Besonders für den Geometrieblock ist festzuhalten: Das Legacy-`Foliation`-Modul trägt nur einen kleinen Präfix-/Längen-Kern, während die eigentliche Foliationssemantik dort faktisch in `SpacetimePaths` über $\tau := \text{length}$, Schichten Σ_n und Adaptedness-Begriffe liegt. Im Refactor wird dies *nicht* roh portiert: `Geometry/Foliation` fixiert nur die kanonische Blätterung der bereits vorhandenen Levelstruktur, während Weltlinien-, Tube- und Kausalbegriffe Consumer in `Geometry/SpacetimePaths` bleiben.

6.5 P27-Archivpolitik für CNNA-Altpfade

Nach grünem P27 werden die alten CNNA-Altpfade *nicht* physisch gelöscht, sondern gesammelt nach `legacy_sources/CNNA_v0.100_unstable` verschoben. Dazu zählen insbesondere alte Bridge-, Core-, OQS-, AQFT- und Integration-Pfade des bisherigen Schritt-1-Strangs sowie die zwischenzeitlichen Top-Level-Ordner `Gates`, `Meta` und `Seeds`, soweit sie durch den neuen Pillar-A-Pfad ersetzt wurden. Die Archivspur bleibt damit als Regressionsorakel und historische Referenz erhalten, liegt aber außerhalb des aktiven Build-Pfads und darf weder neue Fachlogik aufnehmen noch als aktiver Import- oder Ersatzpfad zurückwirken. Nach der in S6c vorgesehenen Handoff-Neustrukturierung gilt dies zusaetzlich fuer jede flache Restablage alter Handoff-Sammelmodule; der aktive Root-Build bleibt auf `CNNA/BuildAll` → `CNNA/Notation` → `CNNA/PillarA/BuildAll` reduziert, waehrend `PillarA/Structural/*` und `PillarA/ScalarEmergence/*` als A-interne Nicht-Handoff-Unterbaeume innerhalb desselben aktiven Pillars sichtbar sind und `CNNA/PillarB` bis `CNNA/PillarE` im aktiven Baum nur `BuildAll.lean`-Stubs tragen.

7 Operative Refactor- und Strengthening-Arbeitsordnung

Der Hauptstrang bis `PillarAStep1Closed` ist zweistufig.
Ontische Zusatzregel für P1–P6: Der ToC-Kern wird zuerst als ideales unendliches Substrat modelliert. Darauf folgen eine adressierte Präsentation dieses Ideals und eine strikt strukturtreue Äquivalenz zwischen abstrakter und adressierter Präsentation. Erst danach wird die reale endliche ToC-Schicht als gerichtete Approximantenfamilie dieses Idealobjekts formuliert. Weder $\mathrm{REAL}_\infty = \mathrm{IDEAL}_\infty$ noch $\mathrm{IDEAL}_\infty^{\text{abstrakt}} = \mathrm{IDEAL}_\infty^{\text{adressiert}}$ werden im Plan als rohe Typgleichheit verstanden;

Ziel sind theorematische Rekonstruktions- bzw. Äquivalenzaussagen über stabile endliche Ausschnitte, gerichtete Übergänge und bidirektionale Transfers. Die in P1 bereits eingeführten unendlichen Fäden sind dabei Foundation-Vorstruktur; `InfiniteCarrier` in P23 ist *nicht* der erste Ort der Unendlichkeit, sondern ein später, netz- und handofffähiger Trägerrahmen über bereits geschlossenem A-Kern.

Stufe A: Wurzel → früher A-Kernschluss (bis `RegularizationClosure`)

```
SubstrateClass -> MatrixNorms -> Interfaces -> Determinism -> ToC/Contract
-> ToC/Addressing -> ToC/IdealAddressEquiv -> ToC/Concrete -> Finite/CutSpec
-> RegionCore -> BoundaryPorts -> Approximant -> DirichletLaplacian -> DtN
-> DtNStabilized -> BranchPatch -> ComplementSectorFamily -> SectorSplit ->
BranchingWitness -> SelectedBranching + UVSpectralSelector -> BranchingSelector
-> ParameterClosure -> RegularizationClosure
```

Stufe B: vom geschlossenen frühen A-Kern zur Abgangswelle

```
SectorSplit -> RegionNet
```

parallel dazu

```
DtN + SectorSplit -> GeneralizedDtN -> MultiSchur
```

und danach

```
InfiniteCarrier -> SectorExport -> Step1StrongCore -> Step1MathData -> ABHandoffStrong
-> PillarAStep1Closed
```

Harte Klassifikation

- **Unverzichtbarer Hauptstrang:** Nr. 1–12, 14–29, 36–40. Modul 29 (`InfiniteCarrier`) ist notwendig, aber spät und nur aus bereits geschlossenen A-Strukturen abgeleitet.
- **Bedingter Support:** Nr. 13 (`Selection`) bleibt auch nach grünem P10 ein Supportmodul: es ist nun zwar derived-only geschlossen, wird aber nur dann aktiv als Dependency des Hauptstrangs genutzt, wenn die Operatorstufe eine echte Teilstruktur-Auswahl benötigt.
- **Flankierender Support:** Nr. 30–35 (`SectorChannels`, `SectorSysEnv`, `RelativeEntropyFlow`, `Geometry/LCPMeasure`, `Geometry/Foliation`, `Geometry/SpacetimePaths`) sind Nebenstrangmodule; sie dürfen den Hauptpfad nicht rückwärts beeinflussen.
- **Konstruktionsregel:** `SectorExport`, `Step1MathData` und `ABHandoffStrong` dürfen keine Sammelstellen roher Einzelinputs werden; sie sind als Komposition bereits geschlossener Teilstränge zu formulieren.
- **Backreaction-Regel:** Rueckwirkung darf nie ad hoc modelliert werden. Ein Rueckwirkungspfad existiert genau dann, wenn ein Satz eines tieferen Pfeilers erst nach Emergenz eines hoeheren Pfeilers schliessbar wird; dann darf die Rueckwirkung ausschliesslich ueber einen expliziten `Input`-Handoff dieses tieferen Pfeilers laufen. Direkte Rueckgriffe auf spaetere Pillar-Module oder deren `Output`-Objekte sind unzulaessig.
- **Emergenzstatus-Regel:** Aussagen tieferer Pfeiler sind semantisch nach Emergenzstatus zu lesen. Insbesondere sind prae-B- und post-B-Lagen nicht stillschweigend zu identifizieren; ohne Subsystemwahl und Split bleibt Pillar A im reinen IDEAL-Modus ohne Backreaction.
- **Branching-Consumer-Regel ab P16:** `BranchingSelector` ist als Consumer von `SelectedBranching` plus `UVSpectralSelector` zu formulieren; rohes `SectorSplitStrong` bleibt im Folgepfad für seine eigenen direkten Consumer wie `RegionNet` oder `GeneralizedDtN` reserviert und darf nicht als Ersatzoberfläche für die bereits geschlossene Witness-/Auswahlstufe dienen.

- **Coupling-Dualpfad-Regel ab P21/P22:** Die Kopplungsseite darf die in P13 explizit geschlossene Trennung von rohem und stabilisiertem Randoperator nicht wieder implizit zusammenziehen. `GeneralizedDtNStrong` bleibt ausschließlich Consumer von `DtNStrong` und `SectorSplitStrong`, muss aber daraus zwei explizite, theorematisch gebundene Operatoroberflächen — raw und stabilized — derselben legalen P21-Provenienz tragen. `MultiSchurStrong` bleibt reiner P22-Consumer von `GeneralizedDtNStrong` und arbeitet operativ nur auf dem stabilisierten Restriktionspfad; der rohe Pfad bleibt als Audit-/Diagnostikoberfläche sichtbar. Ein direkter Closure-Seitenpfad von `GeneralizedDtN`/`MultiSchur` nach `RegularizationClosureStrong` ist kein zulässiger Architekturfix.

8 Menschenlesbare Notation als Parallelstrang

- Für grün geschlossene aktive Frühmodule des Pfads P1–P4 ist grundsätzlich eine kleine modulnahe `Notation.lean`-Datei vorzusehen, sofern das Modul direkte Kernobjekte des aktiven Pfads exponiert; spätere Module folgen derselben Regel bei Bedarf.
- Verwendet wird nur menschenlesbare, möglichst in Mathematik oder Physik etablierte Notation; keine kosmetische Fantasienotation und keine privaten Kürzel für nicht etablierte Begriffe.
- Die Lean-Objekte selbst sollen bereits menschenlesbare, fachlich sprechende Namen tragen; Notation ergänzt einen guten Kernnamen, sie kompensiert keine kryptischen Objektbezeichner. Für Reviewer soll der aktive Pfad auch ohne tiefes Nachschlagen in Lean/Mathlib in seiner fachlichen Grundstruktur lesbar bleiben.
- Die Notation wird erst freigegeben, wenn der zugrunde liegende Objektkern bereits derived-only geschlossen ist.
- Notation ist **scoped** und rein lesbare Makro-Ebene; sie darf keine semantische Verschiebung des Kerns erzeugen. Bevorzugt werden lesbare Aliasnamen und nur dort Symbolnotation, wo diese fachlich wirklich etabliert und für Reviewer unmittelbar interpretierbar ist.
- Für die frühe Schichtungssemantik wird die Carrier-Ebene strikt von verpackten Slice-Objekten getrennt: Σ_k bezeichnet eine Level-k-Schicht als Trägermenge (im Root-Fall insbesondere Σ_0 als Wurzelschicht), während Slice-Objekte wie `rootSlice` oder allgemein `LayerSlice` ausgeschrieben bleiben. Dieselbe Symbolik darf nicht zugleich für Carrier und Slice-Datum verwendet werden; insbesondere wird Ω_0 nicht doppelt für Wurzelschicht und Wurzelslice belegt.
- Für familienparametrische Objekte müssen Notationen ihre Parameter sichtbar oder explizit bindbar lassen; insbesondere darf eine Notation keine Typklasseninstanz schon beim Import erzwingen.
- Nullstellige Notation für noch offene familienparametrische Objekte ist zu vermeiden; insbesondere dürfen Objekte mit Parametern wie `Cell` oder Instanzen wie `[SubstrateClass Cell]` nicht als scheinbar geschlossene Konstanten maskiert werden.
- Notation darf keine abhängigen Level-/Carrier-Transporte oder Äquivalenztransfers verdecken; wo solche Transporte fachlich tragen, bleiben sie explizit im Kern oder werden als eigene Rekonstruktions- bzw. Transfersätze formuliert.
- Jede Refactor-Phase prüft zusätzlich, ob für die neu geschlossenen Objekte die passende lesbare Notation lokal ergänzt werden muss.
- Mit grünem P16 sind in `Sectors/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `BranchingWitness` und `SelectedBranching` freigegeben; spätere Selector-/Closure-Module dürfen diese nutzen, müssen aber Familienparameter sichtbar lassen und dürfen weder zulässige Levelmengen noch Minimalitätsbeweise hinter nullstelliger Symbolik verstecken.

- Mit grünem P17 sind in `Sectors/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `UVSpectralSelector` und `BranchingSelector` freigegeben; diese dürfen die in P17 fachlich tragenden abhängigen Level-/Carrier-Transporte nicht verdecken und keine scheinbare definitorische Gleichheit zwischen Branching- und UV-Seite suggerieren.
- Mit grünem P18 sind in `Closure/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `ParameterClosure` freigegeben; diese müssen die Familienparameter sichtbar lassen, dürfen die aus `DtNStabilized` kommende computable Shift-/Stabilisierungsprovenienz nicht verdecken und keine scheinbar geschlossene Regularisierungsstufe suggerieren, die erst P19 liefert.
- Mit grünem P19 sind in `Closure/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `RegularizationClosure` freigegeben; diese müssen die Familienparameter sichtbar lassen, dürfen keine zweite oder semantisch stärkere Regularisierungslogik suggerieren und die aus `ParameterClosure`/`DtNStabilized` übernommene computable Provenienz weder verdecken noch als definitorisch neue Schicht maskieren.
- Mit grünem P20 sind in `Network/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `RegionNet` und `RegionKind` freigegeben; diese müssen die Familienparameter sichtbar lassen, dürfen weder eine zusätzliche Netzsemantik über `SectorSplitStrong` hinaus suggerieren noch die Provenienz der hellen/Interface/dunklen Projektionen als definitorisch unabhängige Schicht maskieren.
- Mit grünem P21 sind in `Coupling/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `GeneralizedDtN` und `CoupledSectorKind` freigegeben; diese müssen die Familienparameter sichtbar lassen, dürfen weder eine zusätzliche Netz- oder Closure-Semantik über `DtNStrong` und `SectorSplitStrong` hinaus suggerieren noch die blockweise Kopplungsoberfläche als definitorisch unabhängige zweite Operatorstufe maskieren. Falls P22 rückwirkend zusätzliche P21-Strukturen benötigt, dürfen die zugehörigen Aliasoberflächen lokal ergänzt werden, solange dadurch keine `MultiSchur`-Fachlogik in die Notations- oder Kopplungsdatei von `GeneralizedDtN` vorverlagert wird.
- Mit grünem P22 sind in `Coupling/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `MultiSchur` freigegeben; diese müssen die Familienparameter sichtbar lassen, dürfen weder eine zusätzliche Netz-, Closure- noch Handoff-Semantik über `GeneralizedDtNStrong` hinaus suggerieren noch die in P22 explizit getragenen Restriktions-, Gluing- und Schur-Kompositionsdaten als definitorisch unabhängige dritte Operatorstufe maskieren. Die Aliasoberfläche darf insbesondere nicht den Eindruck erwecken, als sei die reduzierte Schur-Seite bereits ein selbständiger Ersatz für spätere Export- oder Netzconsumer.
- Mit grünem P24 sind in `Network/Notation`, `Geometry/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `SectorSysEnv`, `RelativeEntropyFlow`, `LCPMeasure`, `Foliation` und `SpacetimePaths` freigegeben; diese müssen die Familienparameter sichtbar lassen, dürfen weder die System/Interface/Environment-Reindexierung noch die Präfix-/Level-Transporte des Geometrieblocks als definitorisch unabhängige neue Kernschichten maskieren und müssen die Trennung Σ_k als Carrier-Schicht gegenüber ausgeschriebenen Slice-Objekten ausdrücklich wahren; τ- bzw. Σ-Notation darf insbesondere keine zusätzliche Dynamik oder Kausalstruktur bereits in `Foliation` suggerieren, die erst `SpacetimePaths` als Consumer tragen darf.
- Mit grünem P25 sind in `Handoff/Notation` bzw. `PillarA/Notation` lesbare Aliasoberflächen für `SectorExport` und `Step1StrongCore` freigegeben; diese müssen die Familienparameter sichtbar lassen, dürfen weder die zielneutrale Exportfläche von A mit dem gerichteten Handoff verwechseln noch tiefe Provenienz- oder Transportketten als scheinbar definitorisch neue Fachschicht maskieren. Die Aliasoberfläche darf insbesondere nicht den Eindruck erwecken, als enthalte `Step1StrongCore` bereits das volle A→B-Handoff oder als sei `SectorExport` selbst schon ein gerichteter Adapter.
- Mit grünem P26 sind in `PillarA/Notation` lesbare Aliasoberflächen für `Step1MathData`, `ABHandoffStrong`

und `PillarAStep1Closed` global freigegeben; diese müssen die drei Stufen weiterhin klar trennen: `Step1MathData` als volles proof-carrying A-Mathematikpaket, `ABHandoffStrong` nur als gerichteter Adapter `A_to_B`, und `PillarAStep1Closed` als einzig legitimes Endobjekt des geared Pfads bzw. der zwischenzeitlichen dünnen Gate-Stufe. Keine Aliasoberfläche darf diese Trennung verwischen oder der Gate-Stufe eigene Produktionslogik suggerieren. Die modulnahe Spiegelung in `Handoff/Notation` für `ABHandoffStrong` und `PillarAStep1Closed` sowie ihre konsequente Konsumption im Handoff-Pfad wurde mit dem grünen P27-Archiv-Cutover nachgezogen.

- Notationsdateien dürfen keine Importzyklen erzeugen, keine spätere Fachlogik vorwegnehmen und keine theorematisch tragenden Parametrisierungen durch Symbolik überspielen. Für P4 bleibt die sichtbare Review-Oberfläche bewusst auf `root`, `parent`, `children`, `ancestor`, `Prefix`, `cellOf`, `addressOf` konzentriert.

9 Vorabprüfung vor Modulschluss

- alle direkt importierten tragenden Module sind selbst bereits derived-only,
- die Imports sind nicht-zirkulär,
- die Imports sind nicht vorausnehmend,
- der neue Builder verwendet nur die zugelassenen starken Vorgänger und keine losen Rohdaten,
- ein expliziter Abtriebssatz zum nächsten zulässigen Consumer ist formuliert,
- im ToC-Pfad ist geprüft, dass der Idealvertrag nicht durch endliche `Concrete`-Daten vorweggenommen wird und dass `Concrete` den Idealträger nur approximiert, aber nicht definiert,
- vor `ToC/Concrete` ist geklärt, ob der aktive Idealpfad über eine explizite Adressierungsstruktur oder über eine Klasse adressierbarer Ideale läuft; `ToC/Addressing` und `ToC/IdealAddressEquiv` müssen dabei die Transportdaten explizit ausweisen,
- `ToC/IdealAddressEquiv` erhält mindestens Level, Wurzel, Eltern-/Kinder-Struktur, Nichtterminalität, Fäden sowie `refine/coarsen`; spätere `CutSpec`-, Frontier-, Komplement- und Boundary-Daten werden nur auf dieser Basis übertragen,
- ab P5 ist geprüft, dass jede spätere Objektklasse des "Box-Inhalts" – insbesondere Kanten, Schnitte, Approximanten, Rand-/Komplementdaten, Environment-Splits sowie IR/UV-Anteile – nur über diese invarianten Transportdaten identifiziert wird und keine präsentationslokale Ersatz-Kanonizität einführt,
- ab P7 ist für jeden Consumer des endlichen Box-Inhalts der Primärträger explizit festgelegt – abstrakt-konkrete P6-Seite, adressierte Präsentation oder äquivalent rückgeführte Seite – und adressseitige Beschreibungen werden nur als transportierte Präsentationen derselben kanonischen Daten zugelassen; bei abhängigen bzw. indizierten Finite-Objekten sind diese Transporte im Kern explizit durch Rekonstruktions- oder Cast-Sätze auszuweisen und nicht als definitorische Gleichheit zu kaschieren,
- ab P7 ist zusätzlich zu prüfen, dass kanonische adressierte Schnitte des aktiven Finite-Pfads – insbesondere Zylinder- und Teilbaum-Schnitte – über entscheidbare Präfixdaten computable formuliert sind; vermeidbare `noncomputable`-Restquellen im aktiven Box-Pfad sind vor P9 zu beseitigen und nicht als Dauerzustand in spätere Consumer mitzuschleppen,
- ab P9 sind Wrapper- und Projektionsoberflächen des aktiven Box-Pfads grundsätzlich nicht pauschal mit `@[simp]` zu markieren; automatische Aufklappung ist nur für klar stabile Projektionen zulässig und im Zweifel zu vermeiden,

- ab P13 ist zusätzlich zu prüfen, dass Regularisierungs- und Shift-Stufen des aktiven Operatorpfads ihre computable Provenienz explizit ausweisen; analytische `noncomputable`-Hilfsgrößen dürfen separat bestehen bleiben, aber nicht stillschweigend als konsumierte Kernoberfläche von `DtNStabilized` oder seinen Notationsaliasen weitergereicht werden,
- ab P16 ist zusätzlich zu prüfen, dass spätere Selector-, Closure- und Export-Consumer den Branching-Zustand nur über die proof-carrying Oberfläche von `BranchingWitness`/`SelectedBranching` konsumieren; zulässige Branching-Levels, Aktivitätsdiagnostik und Minimalitätsaussagen dürfen nicht erneut aus rohen Sektorcarriern, bloßen Aktivitätszählern oder ad-hoc-Minima auf `SectorSplitStrong` rekonstruiert werden,
- ab P17 ist zusätzlich zu prüfen, dass abhängige Level-/Carrier-Beziehungen zwischen Branching- und UV-Selektoroberfläche explizit theorematisch transportiert und nicht als definitorische Gleichheit kaschiert werden; Notation und Aliasnamen dürfen diese Transporte nicht unsichtbar machen,
- ab P18 ist zusätzlich zu prüfen, dass Closure-Consumer den Branching-/UV-Zustand nur über `BranchingSelectorStrong` und die Stabilisierung nur über die explizit ausgewiesene computable Oberfläche von `DtNStabilizedStrong`/`ParameterClosureStrong` konsumieren; delegierte Projektionen auf Witness-, Split- oder Rohoperator-Ebene dürfen nicht als semantischer Ersatzpfad für die Closure-Stufe dienen,
- ab P19 ist zusätzlich zu prüfen, dass spätere Closure-, Export- und Handoff-Consumer die abgeschlossene Regularisierungsseite ausschließlich über `RegularizationClosureStrong` konsumieren; Rückgriffe auf `ParameterClosureStrong`, rohe `DtNStabilized`-Projektionen oder ältere OQS-Regularisierungspfade sind dabei nicht als semantisch gleichwertige Ersatzpfade zulässig,
- ab P20 ist zusätzlich zu prüfen, dass direkte Netz-Consumer des Sektorpfads ihre operative Oberfläche ausschließlich aus `SectorSplitStrong` gewinnen; Closure-, Branching-, DtN- oder ältere Bridge-/OQS-Pfade dürfen dort nicht als semantisch gleichwertige Ersatzoberflächen eingeblendet werden, und sektorielle Regions-/Boundary-/Carrier-Projektionen bleiben derived-only Projektionen derselben Netzstufe,
- ab P21 ist zusätzlich zu prüfen, dass Coupling-Consumer ihre operative Oberfläche ausschließlich aus `DtNStrong` und `SectorSplitStrong` gewinnen; `RegionNetStrong`, `RegularizationClosureStrong` oder ältere Bridge-/OQS-Pfade dürfen dort nicht als semantisch gleichwertige Operatorersatzflächen eingeblendet werden. `GeneralizedDtNStrong` hat dabei raw/stabilized als explizit getrennte, theorematisch gebundene Operatoroberflächen derselben legalen P21-Provenienz sichtbar zu tragen; ein semantisch mehrdeutiger Einheitsoperator ist als operative Kernoberfläche zu vermeiden. Falls P22 rückwirkend zusätzliche P21-Strukturen benötigt, sind diese lokal in `GeneralizedDtN` nachzutragen, ohne die Fachgrenze zum `MultiSchur`-Modul zu verwischen,
- im aktiven geared Hauptpfad sind neue `noncomputable`-Produktionsdefinitionen grundsätzlich unzulässig, sofern nicht lokal theorematisch ausgewiesen ist, dass sie mathematisch unvermeidbar sind, keine computable Ersatzformulierung existiert und die konsumierte Consumer- bzw. Exportoberfläche dennoch vollständig computable bleibt,
- ab P23 ist geprüft, dass `InfiniteCarrier` keine neue ontische Unendlichkeitsquelle einführt, sondern nur bereits geschlossene Foundation-/ToC-Unendlichkeitsdaten netzfähig bündelt.
- ab P24 ist zusätzlich zu prüfen, dass flankierende Netzwerk- und Geometrieconsumer ihre operative Oberfläche ausschließlich aus den bereits geschlossenen Supportvorgängern gewinnen: `SectorSysEnvStrong` nur aus `SectorChannelsStrong`, `RelativeEntropyFlowStrong` nur aus `SectorSysEnvStrong`, und der Geometrieblock strikt entlang `LCPMeasure` → `Foliation` → `SpacetimePaths`; dabei darf Σ_k nur Level-Carrier bezeichnen, während Weltlinien-, Tube- und Kausalbegriffe nicht vorzeitig in `Foliation` oder in Notationsaliasen vorweggenommen werden.

- ab P25 ist zusätzlich zu prüfen, dass Export- und Handoff-Consumer ihre operative Oberfläche ausschließlich aus den bereits geschlossenen reifen A-Strängen gewinnen: `SectorExportStrong` nur als Projektion reifer Netz-, Closure- und Coupling-Daten, `Step1StrongCore` nur als kompakte Zusammenfassung bereits geschlossener Teilstränge. Weder flankierende Netzwerk-/Geometrieconsumer noch rohe Vorgänger, Ersatzpfade oder ältere Bridge-/OQS-Handoffs dürfen dort als semantisch gleichwertige Exportoberflächen eingeblendet werden. Insbesondere ist P25 kein Reparaturort für P21/P22: Die Explizitheit raw/stabilized muss bereits in der Coupling-Schicht geschlossen sein und darf im Export nur noch theorematisch sichtbar projiziert, nicht semantisch nachgebessert werden.
- ab P26 ist zusätzlich zu prüfen, dass `Step1MathData` und `ABHandoffStrong` keine Sammelstellen roher Einzelinputs werden: `Step1MathData` bündelt nur das volle proof-carrying A-Mathematikpaket über dem bereits geschlossenen Handoff-Kern, `ABHandoffStrong` bleibt ausschließlich der gerichtete Adapter `A_to_B`, und `PillarAStep1Closed` ist das einzig legitime Endobjekt des geared Pfads. Die zwischenzeitliche dünne Step-1-Gate-Stufe konsumiert nur `PillarAStep1Closed` und enthält keine eigene Produktionslogik mehr.
- ab P27 ist zusätzlich zu prüfen, dass nach erfolgreichem Konsumentenwechsel alte Bridge-, Core-, OQS-, AQFT-, Integration-, Gates-, Meta- und Seeds-Pfade nicht mehr im aktiven Build-Pfad verbleiben, sondern gesammelt nach `legacy_sources/CNNA_v0.100_unstable` verschoben werden; diese Archivspur bleibt reines Regressionsorakel außerhalb des Build-Pfads und darf weder neue Fachlogik aufnehmen noch als aktiver Import- oder Ersatzpfad zurückwirken. Zugleich bleibt aktiv nur der neue derived-only Pillar-A-Pfad, während `PillarB` bis `PillarE` höchstens `BuildAll.lean`-Stubs tragen.
- ab S10 ist zusätzlich zu prüfen, dass Update-, Diagnose-, Symmetrie-, thermische und dynamische Seeds ausschließlich über bereits geschlossene A-Kern- bzw. algebraische Seed-Vorgänger konsumieren; sie dürfen weder Closure- noch Coupling-Logik als Rückkanal benutzen noch B/C/D-Fachsemantik vorwegnehmen.

Folge: Modulschluss ist niemals nur lokal. Er ist stets ein Urteil über die tragfähige Importkette bis zur Wurzel.

10 Exakte Programmierreihenfolge der reinen Pillar-A-Core-Module

Nr.	Modul	Muss lokal erfüllen	Muss wie verzahnen
1	`Foundation/SubstrateClass`	idealer geschichteter Basiskontrakt: Wurzel auf Ebene 0, lokale Endlichkeit, Eltern-/Kinder-Kohärenz, nichtterminale Verzweigung, dynamische Branching-Daten, `refine/coarsen`-Vorstufe, explizite unendliche Fäden; liefert die einzige primitive diskrete Wurzel des aktiven Pfads, aus der spätere Trägerdaten derived-only hervorgehen; die präzise leere-Ursprung-Semantik bleibt als Anschluss an `ToC/Contract` markiert	Wurzel für `MatrixNorms`, `Interfaces` und den idealen ToC-Vertrag

Nr.	Modul	Muss lokal erfüllen	Muss wie verzahnen
2	`Foundation/MatrixNorms`	nur fundamentale Matrix-/Normfakten; mindestens Frobenius-Kern mit Nichtnegativität, Nulltest und Positivität bei $\neq 0$; Regularisierungsbausteine für den aktiven Pfad sind so auszuweisen, dass ihre konsumierte Shift-Stufe explizit computable bleibt und analytische `noncomputable`-Hilfsgrößen nicht stillschweigend in `DtNStabilized` weitergereicht werden	versorgt `DirichletLaplacian`, `DtN`, `DtNStabilized`
3	`Foundation/Interfaces`	kleinste gemeinsame Typschnittstellen über dem Foundation-Kern; exponiert nur mathematisch tragende Schnittstellen und keine bequemen Sammel-APIs oder Labelhalden	versorgt `ToC/Contract`, `RegionCore`, `BoundaryPorts`
4	`Foundation/Determinism`	deterministische Auswahl- und Eindeutigkeitssätze aus Policy-/Key-Daten auf Basis des Foundation-Kerns; liefert kanonische Auswahl als Satz und keine bloßen Hilfskonstruktionen oder Export-Hüllen-Ersatzlogik	versorgt `ToC/Concrete`, `BranchingSelector`, `Step1MathData`
5	`ToC/Contract`	reiner abstrakter Idealvertrag des unendlichen ToC: idealer Träger, kohärente Ketten auf Basis der Foundation-Fäden, präzisierte leere-Ursprung-/Fixpunktsemantik, keine terminale Stufe, keine `Concrete`- oder `Dirichlet`-Importe; trägt noch keine adressierte oder endliche Präsentation	treibt `ToC/Addressing` und später `ToC/Concrete`
6	`ToC/Addressing`	adressierte Präsentation des aktiven Ideals mit Präfix-/Ancestor-Struktur und eindeutiger Zellzuordnung; schließt zunächst die kanonische Zelladressierung, aber noch nicht den gesamten Box-Inhalt; sichtbare Review-Oberfläche bleibt auf `root`, `parent`, `children`, `ancestor`, `Prefix`, `cellOf`, `addressOf` konzentriert; keine ontische Umdefinition des abstrakten Vertrags	versorgt `ToC/IdealAddressEquiv`, `Finite/CutSpec`

Nr.	Modul	Muss lokal erfüllen	Muss wie verzahnen
7	`ToC/IdealAddressEquiv`	strikt strukturtreue Äquivalenz zwischen abstraktem und adressiertem IDEAL$_\infty$: `encode`, `decode`, Links-/Rechtsinverse und Erhaltung von Level, Wurzel, Eltern/Kindern, Fäden, `refine/coarsen`; ab hier müssen spätere Objektklassen über P5-invariante Transportdaten identifiziert werden	kanonische Transferbrücke für `ToC/Concrete`, `Finite/CutSpec` und spätere Transfersätze
8	`ToC/Concrete`	REAL-Approximantenfamilie des idealen ToC mit gerichteten Übergängen in `L_max`; konkrete endliche Realisierung darf lokal zunächst statisches `b` verwenden, bleibt aber Consumer der Äquivalenzschicht und der Determinismus-Sätze; liefert den ersten `ToCStrong`-Kern mit policy-keyed Trunkierungen, auf dem spätere Box-Objekte aufsetzen, statt präsentationslokale Ersatzrepräsentationen als Primärträger einzuführen	versorgt `Finite/CutSpec`, `RegionCore`, `Approximant`, `UVSpectralSelector`
9	`Finite/CutSpec`	allgemeine endliche Schnittspezifikation über P6-Approximanten und adressierte bzw. äquivalent rückgeführte ToC-Daten; umfasst mindestens Trunk-, Fenster-, Zylinder-/Teilbaum- und adaptive Frontier-Schnitte; legt die kanonischen Schnitte des Box-Inhalts fest und weist explizit aus, auf welcher Seite der Schnitt primär lebt, während adressseitige Beschreibungen nur transportierte Präsentationen desselben Schnitts bleiben; adressierte Zylinder-/Teilbaum-Schnitte sind im aktiven Pfad über entscheidbare Präfixdaten computable zu formulieren und nicht bloß als tolerierte `noncomputable`-Restfälle stehen zu lassen	treibt `RegionCore`, `BoundaryPorts`, `Approximant` und spätere Environment-Splits
10	`Finite/RegionCore`	kleinster Regionskern aus `CutSpec` und ToC-Daten; Primärträger von Region und Innenbereich ist explizit festgelegt und wird nicht stillschweigend mit seiner adressierten Präsentation identifiziert	treibt `BoundaryPorts`, `Approximant`, `BranchPatch`

Nr.	Modul	Muss lokal erfüllen	Muss wie verzahnen
11	`Finite/BoundaryPorts`	kanonische Randport-Extraktion und damit erste kanonische Randmarkierung der endlichen Box; Randdaten werden aus dem in `CutSpec/RegionCore` fixierten Primärträger gewonnen und nicht über adresslokale Ersatz-Kanonizität eingeführt; die Oberfläche soll auch den für spätere Consumer benötigten Innenbereich so exponieren, dass kein Seitenimport zurück nach `RegionCore` nötig wird	liefert Randdaten für `SectorSplit`
12	`Finite/Approximant`	endliche ToC-Abschneidung mit Herkunftssätzen aus `CutSpec`; ist Consumer des in P6 bereitgestellten Approximantenkerns und des in `CutSpec/RegionCore/BoundaryPorts` kanonisch fixierten endlichen Box-Inhalts; fixiert diesen zum gegebenen Cutoff, ohne einen zweiten `Concrete`-Kern oder ad-hoc Determinismuspfad aufzubauen	treibt `Selection`, `DirichletLaplacian`, `BranchPatch`, Selektoren
13	`Finite/Selection`	ausgewählte endliche Teilstruktur	bleibt Supportmodul, kein Alternativpfad
14	`Finite/DirichletLaplacian`	Symmetrie-, Nichtnegativitäts- und Blockzerlegungssätze über dem bereits kanonisch fixierten endlichen Box-Inhalt; der operative Indexträger ist ein gebundener endlicher Box-Träger über allen Levels $\leq$ Cutoff, nicht bloß eine Topschicht und nicht ein ungebundener globaler Träger; Rand-/Innen-Partition und die zugehörigen Blockmatrizen sind als derived-only Folge des Approximantenkerns explizit auszuweisen; analytische Energieterme sind so zu formulieren, dass keine vermeidbare `noncomputable`-Definition in den aktiven Pfad zurückkehrt	einziger direkter Motor für `DtN`; `Selection` darf nur bei echter Teilstruktur-Auswahl konsumiert werden und bleibt sonst unbenutzt
15	`DtN/DtN`	binärer DtN-Kern aus `DirichletLaplacianStrong`	treibt `DtNStabilized` und später `GeneralizedDtN`

Nr.	Modul	Muss lokal erfüllen	Muss wie verzahnen
16	`DtN/DtNStabilized`	expliziter Shift-/Regularisierungssatz; trennt rohen Randoperator, symmetrisierte Hilfsstufe und primäre stabilisierte Operatoroberfläche explizit und konsumiert nur eine computable Shift-Stufe aus `MatrixNorms`, sodass keine vermeidbare `noncomputable`-Restquelle in den aktiven Pfad zurückkehrt	versorgt `ParameterClosure`, `RegularizationClosure`, `SectorExport`
17	`Sectors/BranchPatch`	rein ontische lokale Patch-Struktur	Vorstufe für `SectorSplit`
18	`Sectors/ComplementSectorFamily`	Komplementfamilie aus ToC-/Regionsdaten; erste kanonische Umgebung/Komplementseite der Box	zweiter Input für `SectorSplit`
19	`Sectors/SectorSplit`	helle/interface/dunkle Zerlegung, geometrisch	treibt `RegionNet`, `BranchingWitness`, `GeneralizedDtN`, `SectorChannels`, `SectorSysEnv`
20	`Sectors/BranchingWitness`	proof-carrying Witness-Stufe für zulässige Branching-Level über `SectorSplitStrong`; rekonstruiert eine kanonische Kandidatenmenge samt Admissibility, ohne Umgebungsdiagnostik oder Sektortrennung neu aufzubauen	treibt `SelectedBranching`; spätere Selectoren dürfen Witness-Daten nicht durch Rohzugriffe auf `SectorSplitStrong` ersetzen
21	`Sectors/SelectedBranching`	kanonische Auswahl eines zulässigen Branching-Levels mit Minimalitätsbeweis über der Witness-Stufe	primärer Branching-Input für `BranchingSelector`
22	`Sectors/UVSpectralSelector`	vor-operatorischer UV-/Spektralselektor; markiert den UV-tail nur als derived-only Folge der kanonischen Vorstufen	liefert UV-Selektorinformation an `BranchingSelector` und als Provenienzoberfläche weiter an `ParameterClosure`
23	`Sectors/BranchingSelector`	deterministische/proof-carrying Zusammenführung von `SelectedBranching` und `UVSpectralSelector`	letzter Selektor vor der Closure
24	`Closure/ParameterClosure`	Schließungsparameter aus `DtNStabilized` plus kanonisierter Branching-/UV-Selektoroberfläche; die computable Provenienz der Auswahl bleibt sichtbar	treibt `RegularizationClosure` und `SectorExport`
25	`Closure/RegularizationClosure`	eigentliche Regularisierungs-/Schließungsstufe	letzter strenger Abschluss des frühen Operatorpfads
26	`Network/RegionNet`	Netzstruktur ausschließlich aus `SectorSplitStrong`	Input für `InfiniteCarrier` und `SectorExport`

Nr.	Modul	Muss lokal erfüllen	Muss wie verzahnen
27	`Coupling/GeneralizedDtN`	Mehrsektor-DtN-Kopplung aus `DtNStrong` + `SectorSplitStrong`; trägt raw/stabilized als explizit getrennte, theorematisch gebundene Operatoroberflächen derselben legalen P21-Provenienz sowie die dazugehörigen Restriktions-/Projektionsdaten, ohne Closure- oder Netzersatzpfade einzublenden	treibt `MultiSchur` und `SectorExport`
28	`Coupling/MultiSchur`	Schur-Komposition, Restriktion und Gluing ausschließlich als Consumer der in P21 bereits explizit geschlossenen Kopplungsoberfläche; operativer Schur-Pfad läuft auf dem stabilisierten Restriktionsblock, während der rohe Pfad nur Audit-/Diagnostikoberfläche bleibt	liefert Kopplungsdaten an den Handoff-Strang
29	`Network/InfiniteCarrier`	netz- und handofffähiger unendlicher Trägerrahmen, noch ohne AQFT-Quasilokalität; *nicht* erste Einführung von Unendlichkeit, sondern spätere Bündelung bereits vorhandener Foundation-/ToC-Unendlichkeitsdaten	spät ableiten; versorgt `SectorExport`, `Step1MathData`
30	`Network/SectorChannels`	kanalartige Sektorverbindungen	Supportconsumer von `SectorSplit` bzw. `GeneralizedDtN`
31	`Network/SectorSysEnv`	System/Environment-Zerlegung über dem Sektornetz	versorgt `RelativeEntropyFlow`
32	`Network/RelativeEntropyFlow`	entropische Flussbuchhaltung	späteres Supportmodul; kein Rückeinfluss auf den Kernpfad
33	`Geometry/LCPMeasure`	adressnahe Geometriestufe für längste gemeinsame Präfixe bzw. zellnahe Nähedaten	flankierender Support für spätere Geometrie- und Analysepfade
34	`Geometry/Foliation`	kanonische Blätterung des adressierten bzw. über P5 transportierten Trägers nach Level; definiert gebündelten Träger, Foliationsfunktion τ, Blätter Σ_n als Level-n-Trägermengen sowie elementare Sätze über Blattzugehörigkeit und Gleichheit bei gleicher Schicht unter Präfixrelation; Slice-Objekte bleiben davon begrifflich getrennt und werden weiterhin ausgeschrieben geführt; führt keine zusätzliche Dynamik oder Kausalstruktur ein	geometrische Vorstufe für `SpacetimePaths` und flankierender Support für spätere Analysepfade
35	`Geometry/SpacetimePaths`	pfadartige Weltlinien-/Raumzeitdarstellung über dem adressierten Kern	Support für spätere Pfad- und Flussanalysen

Nr.	Modul	Muss lokal erfüllen	Muss wie verzahnen
36	`Handoff/SectorExport`	vollständiger A-Export aus reifen A-Daten	projiziert nur reife A-Daten und nimmt keine B-Struktur vorweg
37	`Handoff/Step1StrongCore`	kompakte starke Zusammenfassung des Kernstrangs	treibt `Step1MathData`, `ABHandoffStrong`
38	`Handoff/Step1MathData`	volles proof-carrying Mathematikpaket	bündelt die gesamte A-interne Kraftübertragung
39	`Handoff/ABHandoffStrong`	verbreiterter, reifer A→B-Handoff	eigentliche Abgangswelle
40	`Handoff/PillarAStep1Closed`	Endobjekt / Endtheorem des geared Pfads	einzig legitimer Input der zwischenzeitlichen dünnen Gate-Stufe und aktiver Root-Endpunkt nach P27

11 Refactor-Phasen P0–P27 (Schritt 1)

Phase	Module / Bereich	Pflichtresultat
P0	alle in Abschnitt 9 geführten reinen Pillar-A-Core-Module; zusätzlich `DirichletLaplacianBridge`, `StrongLegacyAdapters`, `RegularizationClosure_old`; Inventar, Zielpfade, Freeze; Altmodule markieren	vollständige Klassifikation aller reinen Pillar-A-Core-Module inklusive `ToC/Addressing`, `ToC/IdealAddressEquiv`, `Finite/CutSpec` und der Geometriemodule; alter Pfad nur noch Regressionsorakel
P1	`Foundation/SubstrateClass`, `Foundation/MatrixNorms`; begleitend `Foundation/Notation`	fundamentale Wurzel steht unter `PillarA/Foundation`; `SubstrateClass` trägt bereits den idealen geschichteten Kern des $\texttt{IDEAL}_\infty$-ToC und markiert die einzige primitive diskrete Wurzel des aktiven Pfads; `MatrixNorms` ist kein Stub mehr und hält die vom aktiven Pfad konsumierte Regularisierungsstufe explizit computable, statt analytische `noncomputable`-Hilfsgrößen ungeprüft nach vorne weiterzureichen
P2	`Foundation/Interfaces`, `Foundation/Determinism`; begleitend `Foundation/Notation`	Typschnittstellen und Determinismus-Sätze stehen getrennt *über dem bereits geschlossenen Foundation-Kern*; `Interfaces` exponiert nur mathematisch tragende Schnittstellen, `Determinism` liefert kanonische Auswahl als Satz; keine Bool-Audit-Ersatzschlüsse, keine Labelhalden, Policy-/Key-Bindung ist explizit
P3	`ToC/Contract`; begleitend `ToC/Notation`	reiner abstrakter Idealvertrag des ToC ohne spätere Importe; der ideale unendliche Träger wird *auf Basis des P1-Substratkerns* und der typisierten P2-Schnittstellen präzisiert, trägt die Fixpunktrolle und schließt insbesondere die leere-Ursprung-/Grenzsemantik; P3 trägt den abstrakten Gegenstand, aber noch keine adressierte oder endliche Präsentation; der grüne P3-Stand bleibt bestehen und wird nicht zurückgenommen

Phase	Module / Bereich	Pflichtresultat
P4	`ToC/Addressing`; begleitend `ToC/Notation`	adressierte Präsentation des aktiven $\texttt{IDEAL}_\infty$ steht; Zellen sind adressartig eindeutig und damit kanonisch referenzierbar, ohne den abstrakten Vertrag zu ersetzen; die sichtbare P4-Oberfläche konzentriert sich auf `root`, `parent`, `children`, `ancestor`, `Prefix`, `cellOf`, `addressOf`; die Kanonizität des übrigen Box-Inhalts wird hier noch nicht behauptet
P5	`ToC/IdealAddressEquiv`	strikt strukturtreue Äquivalenz zwischen abstraktem und adressiertem $\texttt{IDEAL}_\infty$ steht; `encode`/`decode`, Inversen und Kernerhaltungssätze sind explizit formuliert; jede spätere Objektklasse darf nur über diese P5-invarianten Transportdaten identifiziert werden
P6	`ToC/Concrete`; begleitend `ToC/Notation`	REAL-Approximantenfamilie des idealen ToC ohne Rückimport aus `Finite/DtN`; `Concrete` bleibt familienparametrischer Consumer der Äquivalenzschicht und der Determinismus-Sätze; abhängige Level-/Carrier-Daten und Äquivalenztransfers werden nicht durch Notation verdeckt; erster `ToCStrong` und Startpunkt späterer kanonischer Identifikationen des endlichen Box-Inhalts
P7	`Finite/CutSpec`	allgemeine endliche Schnittspezifikation steht; mindestens Trunk-, Fenster-, Zylinder-/Teilbaum- und adaptive Frontier-Schnitte sind formulierbar; die Schnitte werden als Consumer der P6-Approximantenfamilie formuliert, der Primärträger der Schnittdaten ist explizit ausgewiesen, und adressseitige Beschreibungen bleiben transportierte Präsentationen derselben Schnitte; notwendige Rückführungen entlang der P5/P6-Äquivalenz sind auch für indizierte Schnittobjekte explizit theorematisch auszuweisen; adressierte Zylinder-/Teilbaum-Schnitte sind dabei im aktiven Pfad über entscheidbare Präfixdaten computable formuliert und nicht bloß tolerierte `noncomputable`-Restfälle; seitlicher Rest/Environment-Split ist architektonisch vorbereitet und als kanonischer Schnitt des Box-Inhalts formulierbar

Phase	Module / Bereich	Pflichtresultat
P8	`Finite/RegionCore`, `Finite/BoundaryPorts`	endlicher Regionskern und Randport-Extraktion stehen getrennt und ohne Vorgriff auf den Operatorpfad; Importe laufen nur über den neuen Foundation-/ToC-Pfad; Primärträger von Region, Innenbereich und Rand ist explizit festgelegt, und adressseitige Beschreibungen werden nur als durch P5/P6 abgesicherte Präsentationen derselben Box-Daten zugelassen; `BoundaryPorts` exponiert dabei auch den für spätere Consumer benötigten Innenbereich, sodass der grüne P8-Stand als geschlossene konsumierbare Finite-Kernstufe für P9 bestehen bleibt
P9	`Finite/Approximant`	`ApproximantStrong` steht als kanonische endliche Abschneidung aus `CutSpec` über einem bereits in P6 bereitgestellten Approximantenkern; `ApproximantStrong` ist damit Consumer und kein zweiter `Concrete`-Kern; es konsumiert den in `CutSpec/RegionCore/BoundaryPorts` kanonisch fixierten endlichen Box-Inhalt, statt Region oder Rand erneut aufzubauen; dabei soll insbesondere kein Seitenimport zurück nach `RegionCore` nötig werden, sofern die konsumierbare `BoundaryPorts`-Oberfläche den benötigten Innenbereich bereits trägt; kein Rückgriff auf Legacy-Interfaces oder ad-hoc Determinismus-Hilfen; der endliche Box-Inhalt ist damit für den gewählten Cutoff kanonisch fixiert; die modulare Oberfläche bleibt nach Bereinigung anfänglicher Wrapper-Automatik explizit, und pauschale `@[simp]`-Aufklappung ist im aktiven Pfad zu vermeiden
P10	`Finite/Selection`	`SelectionStrong` steht als explizit isoliertes Supportmodul über `ApproximantStrong`; kein Alternativpfad, keine Rekonstruktion von `Concrete/RegionCore/BoundaryPorts`, keine neuen `noncomputable`-Restquellen, klassischen Hilfen oder pauschalen `@[simp]`-Rückfälle; modulnahe Notation bleibt parameter-sichtbar und rein lesbar
P11	`Finite/DirichletLaplacian`; begleitend modulnahe Notation	`DirichletLaplacianStrong` steht und konsumiert nur die minimalen Foundation-Matrixfakten sowie den bereits kanonisch fixierten endlichen Box-Inhalt; der zugrunde liegende Operatorträger wird dabei als gebundener endlicher Box-Träger über allen Levels $\leq$ Cutoff explizit fixiert, sodass Rand-/Innen-Blockzerlegung und der Übergang zu P12 nicht auf eine einzelne Topschicht oder auf einen ungebundenen Globalträger ausweichen; `Selection` darf nur bei echter Teilstruktur-Auswahl zusätzlich konsumiert werden und bleibt sonst unbenutzt; einzige direkte Motorstufe für den binären DtN-Kern

Phase	Module / Bereich	Pflichtresultat
P12	`DtN/DtN`; begleitend modulnahe Notation	binärer DtN-Kern ist isoliert; `DtN` importiert kein `RegionNet`, führt mit `InteriorInverse`, `boundaryOperator`, `interiorSolve` und expliziten Abtriebssätzen eine proof-carrying Oberfläche mit und bleibt ohne neue pauschale `@[simp]`-Flächen oder neue `noncomputable`-Restquellen strikt Consumer von `DirichletLaplacianStrong`
P13	`DtN/DtNStabilized`	`DtNStabilizedStrong` schließt den frühen Operatorpfad nach vorn, konsumiert dabei nur den neuen P12-Pfad unter `CNNA/PillarA/DtN/DtN.lean`, trennt rohen Randoperator und symmetrisierte Hilfsstufe explizit und hält die primäre Stabilisierung über eine explizit computable Shift-Stufe; ein Rückfall auf `CNNA/OQS/DtN.lean` oder eine neue `noncomputable`-Kontamination des aktiven Pfads ist ausgeschlossen
P14	`Sectors/BranchPatch`, `Sectors/ComplementSectorFamily`	ontische Patch- und Komplementvorstufe steht separat und ohne spätere AQFT-/Handoff-Last; die Umgebung/Komplementseite wird erstmals kanonisch familienfähig, und die Datenoberfläche unterscheidet explizit zwischen root-zentriertem Ausschnitt ohne äußere Umgebung und Fenstern mit echter Umgebung/Komplementseite, statt beide Fälle erst implizit in späteren Consumern zu vermischen
P15	`Sectors/SectorSplit`; begleitend modulnahe Notation	`SectorSplitStrong` steht als geometrisch-ontische Sektorzerlegung der nun kanonisch markierten Box-Daten; die Zerlegung macht insbesondere die Präsenz bzw. Abwesenheit einer äußeren Umgebung theorematisch unterscheidbar, ohne daraus schon im Kernpfad eine physikalische Symmetrie- oder Brechungsdiagnostik zu machen; dabei ist explizit darauf zu achten, dass `SectorSplit` die in `ComplementSectorFamilyStrong` bereits gewonnene kanonische Außenseite konsumiert und keine konkurrierende Umgebungsdiagnostik erneut aufbaut, auch dann nicht, wenn die Regions-/Boundary-Sicht dieser Außenseite derived-only über `RegionCore.ofCutSpec` und `BoundaryPorts.ofRegion` rekonstruiert wird

Phase	Module / Bereich	Pflichtresultat
P16	`Sectors/BranchingWitness`, `Sectors/SelectedBranching`; begleitend modulnahe Notation	Witness- und Auswahlstufe stehen proof-carrying und ohne zusätzliche Seiteneingänge; `BranchingWitnessStrong` rekonstruiert eine kanonische Menge zulässiger Branching-Levels, `SelectedBranchingStrong` schließt darüber eine Minimalwahl, und beide konsumieren `SectorSplitStrong` direkt, ohne Umgebungsdiagnostik, Sektortrennung oder Branching-Minimalität aus Rohdaten erneut aufzubauen; modulnahe Notation bleibt parameter-sichtbar und rein lesbar
P17	`Sectors/UVSpectralSelector`, `Sectors/BranchingSelector`	Selektorstufe ist deterministisch/proof-carrying und treibt exklusiv die Closure-Parameter; `BranchingSelector` konsumiert die bereits kanonisierte Branching-Oberfläche aus `SelectedBranchingStrong` zusammen mit `UVSpectralSelector` und ersetzt diese nicht durch rohe Aktivitätszähler oder direkten `SectorSplitStrong`-Zugriff; IR/UV-Anteile bleiben auch hier nur abgeleitete kanonische Markierungen, und der UV-tail wird als cutoff-abhängige, aus den kanonischen Vorstufen hergeleitete Provenienzoberfläche exportfähig gemacht, ohne bereits einen Analyse- oder Dynamiksatz zu behaupten; abhängige Level-/Carrier-Beziehungen zwischen Branching- und UV-Seite sind explizit theorematisch zu transportieren und nicht als definitorische Gleichheit zu kaschieren
P18	`Closure/ParameterClosure`	`ParameterClosureStrong` ist harter Consumer von `DtNStabilized` plus kanonisierter Branching-/UV-Selektoroberfläche; die aus P13 kommende computable Provenienz der Shift- und Stabilisierungsdaten darf dabei nicht verlorengehen, und analytische Hilfsgrößen sind nur zulässig, wenn die exportierten Closure-Parameter weiterhin als operative Oberfläche computable bleiben
P19	`Closure/RegularizationClosure`	`RegularizationClosureStrong` schließt die frühe Minimalstrecke des A-Kerns und führt keine neue nichtkonstruktive Regularisierungsschicht ein, sondern schließt explizit an die bereits ausgewiesene computable Shift-/Closure-Provenienz aus P13/P18 an

Phase	Module / Bereich	Pflichtresultat
P20	`Network/RegionNet`; begleitend modulnahe Notation	`RegionNetStrong` ist reine Netzstufe über `SectorSplitStrong`, bündelt helle/Interface/dunkle Regions-, Boundary- und Carrier-Oberflächen nur als derived-only Projektionen derselben kanonischen Sektorquelle und führt keine Closure-, Branching-, `DtN`- oder Altpfade als semantische Ersatzoberflächen ein; modulnahe Notation für `RegionNet` und `RegionKind` bleibt parameter-sichtbar und rein lesbar
P21	`Coupling/GeneralizedDtN`; begleitend modulnahe Notation	`GeneralizedDtNStrong` koppelt Operatorpfad und Sektorstruktur strikt als Consumer von `DtNStrong` und `SectorSplitStrong`, führt sektorielle Restriktions-/Carrier-/Boundary-/Interior-Projektionen sowie blockweise Randoperator-Oberflächen nur als derived-only Kopplungsdaten derselben Stufe und blendet weder `RegionNetStrong`, noch Closure-, noch ältere Bridge-/OQS-Pfade als Operatorersatzflächen ein; zusätzlich schließt P21 raw/stabilized als explizit getrennte, theorematisch gebundene Operatoroberflächen derselben legalen Coupling-Provenienz und stellt die zugehörigen Restriktions-/Projektionsdaten bereit, ohne `MultiSchur`-Fachlogik vorwegzunehmen; modulnahe Notation für `GeneralizedDtN` und `CoupledSectorKind` bleibt parameter-sichtbar und rein lesbar und darf diese Dualität nicht wieder semantisch verwischen
P22	`Coupling/MultiSchur`; begleitend modulnahe Notation	`MultiSchurStrong` liefert Restriktion, Gluing und reduzierte Schur-Komposition als echten Folgeconsumer von `GeneralizedDtNStrong`; etwaige rückwirkend in P21 ergänzte Hilfsstrukturen verbleiben fachlich im Kopplungsvorgänger, während die Schur-Fachlogik ausschließlich in `MultiSchur` selbst liegt; der operative Interface-/Schur-Pfad arbeitet ausschließlich auf dem in P21 explizit geschlossenen stabilisierten Restriktionsblock, während der rohe Pfad als Audit-/Diagnostikoberfläche sichtbar bleibt; modulnahe Notation für `MultiSchur` bleibt parameter-sichtbar und rein lesbar
P23	`Network/InfiniteCarrier`, `Network/SectorChannels`	späte Träger-/Kanalstufe steht; `InfiniteCarrier` bleibt spät abgeleitet, bündelt nur bereits vorhandene Foundation-/ToC-Unendlichkeitsdaten und blockiert den Frühpfad nicht; die Bündelung ist dabei so zu formulieren, dass gerichtete Übergänge und stabile endliche Ausschnitte für spätere Rekonstruktions- bzw. Limesanalysen exportierbar bleiben, ohne bereits AQFT-Quasilokalität oder Typ-III-Aussagen vorwegzunehmen

Phase	Module / Bereich	Pflichtresultat
P24	`Network/SectorSysEnv`, `Network/RelativeEntropyFlow`, `Geometry/LCPMeasure`, `Geometry/Foliation`, `Geometry/SpacetimePaths`; begleitend modulnahe Notation	spätere Support-Netzwerkschicht und adressnahe Geometriestufe stehen ohne Rückeinfluss auf den Kernpfad; `SectorSysEnvStrong` bleibt derived-only Reindexierung über `SectorChannelsStrong`, `RelativeEntropyFlowStrong` reine entropische Flussbuchhaltung über `SectorSysEnvStrong`; innerhalb des Geometrieblocks wird zuerst `LCPMeasure`, dann `Foliation` als kanonische Level-Blätterung und erst danach `SpacetimePaths` als Consumer dieser Blätterung geschlossen; dabei bezeichnet Σ_k die jeweilige Level-Schicht als Carrier, während Slice-Objekte weiterhin ausgeschrieben bleiben; Weltlinien-, Tube- und Kausalbegriffe gehören nicht in `Foliation`, sondern erst in `SpacetimePaths`; die gesamte Schicht bleibt analytisch bzw. geometrisch flankierend
P25	`Handoff/SectorExport`; begleitend modulnahe Notation; `Handoff/Step1StrongCore`	`SectorExportStrong` projiziert nur reife A-Daten aus dem bereits geschlossenen Netz-, Closure- und Coupling-Pfad auf eine zielneutrale Exportoberfläche und wird nicht zur Sammelstelle roher Inputs; der Export hält die Provenienz seiner operativen Daten explizit sichtbar, konsumiert die Regularisierungsseite nur über `RegularizationClosureStrong` und nicht rückwärts über `ParameterClosureStrong` oder ältere Regularisierungspfade, damit spätere Analysepfade nicht auf informelle Rekonstruktionen angewiesen sind und operative Exportdaten von bloß analytischen Hilfsdaten unterschieden bleiben; zugleich repariert P25 keine P21/P22-Unschärfe mehr nachträglich, sondern projiziert nur die dort bereits explizit geschlossene raw/stabilized-Dualität theorematisch sichtbar nach außen; `Step1StrongCore` fasst darauf aufbauend nur bereits geschlossene Teilstränge kompakt zusammen und nimmt weder einen gerichteten Handoff noch Gate-Logik vorweg

Phase	Module / Bereich	Pflichtresultat
P26	`Handoff/Step1MathData`, `Handoff/ABHandoffStrong`, `Handoff/PillarAStep1Closed`; zwischenzeitliche dünne Gate-Stufe; global freigegebene Aliasoberfläche in `PillarA/Notation`	`Step1MathData` bündelt das volle proof-carrying A-Mathematikpaket über `Step1StrongCore`, `ABHandoffStrong` formuliert darauf ausschließlich den gerichteten Handoff `A_to_B`, und `PillarAStep1Closed` schließt die Abgangswelle als einzig legitimes Endobjekt des geared Pfads; die zwischenzeitliche dünne Gate-Stufe konsumiert ausschließlich `PillarAStep1Closed` und enthält keine eigene Produktionslogik; die globale Aliasoberfläche für diese drei Stufen ist freigegeben, während modulnahe Handoff-Notation und konsequente Konsumption als kleiner Nachtrag vor bzw. mit P27 sauber nachzuziehen sind
P27	abgeschlossener Archiv-Cutover; insbesondere `DirichletLaplacianBridge`, `StrongLegacyAdapters`, `RegularizationClosure_old` sowie weitere alte Core-/OQS-/AQFT-/Integration-/Bridge-Pfade und außerdem `Gates`, `Meta`, `Seeds` nach `legacy_sources/CNNA_v0.100_unstable`; `PillarB`–`PillarE` auf `BuildAll`-Stubs reduziert	alte Pfade sind aus dem aktiven Build-Pfad herausgenommen, bleiben außerhalb des Build-Pfads als Regressionsorakel erhalten und dürfen semantisch nicht mehr als aktive Import- oder Ersatzpfade zurückwirken; aktiv verbleibt nur der neue derived-only Pillar-A-Pfad ausgehend vom ToC

Strengthening-Schritt 2. Der abgeschlossene Refactor hat die Importkette bereinigt, die Altpfade archiviert und den aktiven Baum auf den neuen Pillar-A-Pfad reduziert. Der Code-Audit zeigt aber zugleich: Die Abstraktion über `[SubstrateClass Cell]` ist bereits vorhanden, große Teile der `of*`-Builder-Kette stehen schon, und die eigentlichen Lücken sitzen nicht in einer fehlenden Gesamtarchitektur, sondern in wenigen präzisen Generatorengstellen. Für einen produktiv nutzbaren Generator fehlen deshalb nicht nur Referenzfamilie, Variationsfamilie, `WeightPolicy` und die Schließung der freien Builder-Engstellen `DirichletWeight/InteriorInverse/interfaceInverse`; es fehlt darüber hinaus eine explizite **algebraische A-Wurzel** der endlichen Operatorseite: Adjungierte/Hermitizität, finite Hilbertraumstruktur, endliche Matrix-$*$-Algebra, Kommutator- und Antikommutatorlogik, spektrale Zerlegung, Zustandsraum sowie thermische und unitäre Seeds müssen — soweit sie direkt aus A-generierten endlichen Carriern und Operatoren folgen — bereits in fachlich getrennten A-Modulen bereitgestellt werden und dürfen später nicht in B/C als verdeckter Nachbau wieder auftauchen. Der Legacy-Bestand von REALOQS zeigt gerade in den AQFT-Dateien `StarAlgebra`, `State`, `GNS`, `KMS`, `LocalNet`, `StateNet` und den Derived-Modulen, dass diese Vorstufen später tatsächlich vorausgesetzt werden; Strengthening muss sie daher A-seitig aus der numerischen Pipeline in eine algebraische Generatoroberfläche überführen, ohne volle AQFT-Semantik vorwegzunehmen. Hinzu kommen ein explizites **Computability-Audit** des Generators, eine harte **Fintype/Decidable-Propagationsregel**, eine formale **Directed-Limit**-Infrastruktur für die Lesart $\mathrm{REAL}_\infty \to \mathrm{IDEAL}_\infty$ sowie ausdrücklich eine **A-seitige Diagnosevorbereitung** für Spektraldimension sowie für Symmetrie-, Cutoff- und Limesdiagnostik. Ferner ist eine finite Kanal-/Superoperator-Vorstufe als A-seitiger algebraischer Seed bereitzustellen, soweit sie rein algebraischer Consumer der A-generierten Matrixseite bleibt. Diese Werkzeuge gehören architektonisch in Pillar A, dürfen den Generatorhauptpfad nicht zirkular erweitern und dürfen später weder in B noch in anderen Pfeilern ausgelagert werden. Die folgenden Abschnitte listen daher ausschließlich die für Schritt 2 tatsächlich zu ändernden oder hinzuzufügenden Module, Generatorengstellen, algebraischen Seeds und Abarbeitungsphasen.

Semantische Klarstellungen fuer den Strengthening-Pfad

- **Zum Referenzobjekt:** Das konkrete IDEAL_∞ des Plans ist im Referenzfall kein Cayley-Graph und kein frei schwebendes Diagramm, sondern der einfachste robuste rooted ToC-Fall: ein unendlicher regulaerer Verzweigungsbaum in adressierter ToC-Praesentation. Die zweite legale Familie ist nicht unbekannt, sondern der levelabhaengig verzweigende Baum. Alles Weitere ist Zusatzvariation, nicht stillschweigende Voraussetzung.

- **Zur Mathematik von `SpectralDecomposition`:** Der nichttriviale Inhalt liegt nicht in der informellen Tatsache, dass endliche selbstadjungierte Matrizen diagonalisierbar sind, sondern in der proof-carrying und moeglichst berechenbaren Bereitstellung geordneter Eigenwerte, Projektoren, Orthogonalitaet und Exportsaetzen fuer spaetere Seeds und Handoffs. Seit dem gruenden S8a-Einstieg ist dabei explizit festgelegt, dass die öffentliche Produktivoberfläche auf dem `ExecComplex`-Strang liegt, waehrend jede volle analytische **C**-Diagonalisierung nur als getrennter Spiegelstrang mit Brücke sichtbar werden darf.

- **Zur Solver-Frage:** Ziel des Strengthenings sind keine Gleitkomma- oder Blackbox-Inversionen, sondern exakte endliche Operatorobjekte ueber den im Pfad verwendeten algebraischen Grundkoerpern. Wo volle Konstruktivitaet nicht sauber erzielbar ist, bleibt nur ein lokal gekapselter, whitelist-faehiger Solver-Vertrag zulaessig; numerische Diagnostik und Referenzexperimente sind davon getrennt zu halten.

- **Zur Grenze von Pillar A:** A wird durch die endliche Matrix- und Seed-Seite nicht zum beliebigen Mathematik-Framework. Es liefert nur solche algebraischen, spektralen, thermischen, dynamischen, kanalischen und limesseitigen Samen, die direkt aus A-generierten Objekten derived-only entstehen. Haag-Kastler-Axiome, additive Netzlogik, GNS-/KMS-Theorie, offene-System-Interpretation, lokale Kovarianz und Matter/Gauge bleiben Folgearbeit der spaeteren Pfeiler.

- **Zur Raw/Stabilized-Dualitaet:** Der rohe Pfad wird nicht aus nostalgischen Gruenden mitgefuehrt. Er bleibt als Audit-, Regressions-, Regularisierungs- und Diagnoseflaeche erhalten, damit spaetere Aussagen ueber Stabilisierung, Kontraktion oder Closure-Provenienz nicht nur auf dem geglaetteten Endoperator, sondern auf der kontrollierten Transformation desselben Generatorursprungs beruhen.

- **Zur Lean-/mathlib-Problematik der komplexen algebraischen Seeds:** Die oeffentliche algebraische A-Wurzel darf im Strengthening nicht blind auf die analytisch aufgeruestete **C**-Oberflaeche von Lean/mathlib gelegt werden. Die offiziellen mathlib-Dokumente halten fest, dass `Mathlib/Data/Complex/Basic` die komplexen Zahlen als $\mathbf{R}^2$ modelliert, `Mathlib/Data/Real/Basic` wiederum **R** als Cauchy-Vervollstaendigung rationaler Folgen aufbaut und `Mathlib/Analysis/Complex/Basic` **C** als normiertes Feld registriert. Fuer die A-seitige Generatorwurzel ist gerade diese analytische Aufruestung die problematische Stelle: sie ist fuer spaetere Analyse- und Topologieschichten nuetzlich, zieht aber frueh eine nichtausfuehrbare Infrastruktur in die oeffentliche Seed-Oberflaeche hinein. Dokumentierte Community-Praxis fuer ausfuehrbare algebraische Zahlobjekte ist daher nicht, die analytische Bibliotheksstruktur zu verbiegen, sondern einen eigenen algebraischen Typ mit expliziten Instanzen zu definieren und erst spaeter strukturtreu in die staerkere Theorie zu ueberfuehren; als Lehrmuster dient in *Mathematics in Lean* gerade der Aufbau der Gaussian Integers als eigener Datentyp mit Ringstruktur.[1]

- **Zur Konsequenz fuer den CNNA-Generator:** Ein frueh oeffentlich `noncomputable` gemachter Hermitian-/Hilbert-/$*$-Seed bringt spaeter vor allem abstrakte Existenz- und Transportaussagen, aber keinen belastbaren produktiven Generatorpfad fuer Referenzlauf, Variationslauf, Matrixwirkung, endlichen Zustandsraum oder rohe Rauchtests. Fuer Schritt 2 ist daher nicht ein kosmetischer

[1] Vgl. https://leanprover-community.github.io/mathlib4_docs/Mathlib/Data/Complex/Basic.html, https://leanprover-community.github.io/mathlib4_docs/Mathlib/Data/Real/Basic.html, https://leanprover-community.github.io/mathlib4_docs/Mathlib/Analysis/Complex/Basic.html sowie https://leanprover-community.github.io/mathematics_in_lean/.

Wrapper um **C** zulaessig, sondern eine explizit ausfuehrbare algebraische Wurzel, die spaeter per strukturerhaltender Bruecke nach **C** und in die analytische mathlib-Welt erweitert werden kann.

12 Vorbereitungsblock V0–V5: oeffentlich computable algebraische A-Wurzeln

Der folgende Zusatzblock ist kein alternativer semantischer Pfad neben dem Strengthening, sondern eine *Vorbereitungs- und Reparaturstufe* fuer S1. Er wird genau dann geoeffnet, wenn die oeffentliche algebraische A-Wurzel ueber **C** in Lean in einen diffusen `noncomputable`-Status kippt oder wenn die Erweiterbarkeit nach **C** zwar fachlich gewuenscht, aber im aktiven Generatorpfad noch nicht computable realisiert ist. Der Block ist bewusst so formuliert, dass er spaeter *nicht* weggeworfen werden muss: sein Ziel ist ein ausfuehrbarer algebraischer Kern mit spaeterer strukturtreuer Bruecke ExecComplex $\to$ **C**, nicht ein provisorischer **Q**-Hack ohne Rueckbindung.

Zielbild des Vorbereitungsblocks

Am Ende des Blocks gilt:

- die oeffentliche algebraische A-Wurzel fuer `HermitianStructure`, `FiniteHilbert` und `MatrixAlgebra` ist **computable**;
- die algebraischen Seeds sind definitorisch **nicht** an die analytische **C**-Schicht gebunden, sondern laufen ueber eine kleine explizite Sternring-/Ringoberflaeche;
- `ExecComplex` ist als eigener ausfuehrbarer Datentyp mit expliziten Instanzen vorhanden und dient als erste konkrete Produktivinstanz;
- eine spaetere strukturtreue Erweiterung ExecComplex $\to$ **C** ist bereits vorbereitet und nicht erst ein spaeterer Architekturumbau;
- die spaeteren Seeds konsumieren dieselbe algebraische Wurzel statt lokaler Hilfskonstruktionen oder stillschweigender **C**-Spezialpfade;
- die algebraische A-Wurzel wird explizit als **paralleler Dualstrang** gefuehrt: ein *operativer Produktivstrang* ueber `ExecComplex` und ein *analytischer Spiegelstrang* ueber **C**, verbunden nur ueber die bewiesene Bruecke; der strikte Generatorpfad darf zu keinem Zeitpunkt versehentlich den **C**-Strang als operativen Ausgabekanal konsumieren.

Praezisierung der Koeffizienten- und Parametergrenze

Der Vorbereitungsblock zieht `ExecComplex` bewusst *zunaechst* auf eine ausfuehrbare rationale Repräsentation, etwa $(a, b) \in \mathbf{Q}^2$. Das ist kein semantischer Ausweichtrick, sondern genau die Grenze, an der der produktive Generatorpfad in Lean heute tragfähig bleibt: fuer diskrete Graph-Laplacians, endliche Matrixcarrier und ganzzahlige bzw. rationale Gewichte reichen endliche algebraische Operationen ueber **Q** aus. Die Grenze dieses Startpunkts ist jedoch ebenfalls explizit festzuhalten: sobald spaetere thermische oder policy-nahe Achsen einen genuin reellen Parameter verlangen, darf dieser im produktiven A-Pfad nicht stillschweigend wieder als analytisches **R**-Objekt in die Wurzel einsickern. Insbesondere bleibt die thermische Achse β auf der Generatorseite bis auf weiteres ein sichtbarer rationaler Provenienz- und Variationsparameter; ihre **R**-Interpretation und spaetere physikalische Schließung erfolgen erst ueber die Bruecke in die komplexanalytische Welt und nicht bereits in der algebraischen A-Wurzel.

Operative Designregel

Die Vorbereitung folgt vier harten Regeln:

1. **Kein oeffentlicher analytischer Kurzschluss.** In den S1-Wurzelmodulen `HermitianStructure`, `FiniteHilbert` und `MatrixAlgebra` werden keine normierten, topologischen oder `RCLike`-artigen Klassen als oeffentliche Voraussetzung eingefuehrt.

2. **Expliziter Parallel-Dualstrang.** Die algebraische Wurzel wird von Anfang an als zwei theorematisch gekoppelte, aber semantisch getrennte Straenge gefuehrt: der *operative Produktivstrang* laeuft ueber `ExecComplex` und bleibt oeffentlich computable; der *analytische Spiegelstrang* laeuft ueber **C**, darf intern `noncomputable` sein und ist nur fuer Transfer, Analyse und spaetere Pfeiler zulaessig. Wie beim rohen/stabilisierten DtN darf der strikte Pfad nicht versehentlich auf den falschen Ausgang gestopft werden: Produktionsconsumer duerfen nur den `ExecComplex`-Strang konsumieren, nie die **C**-Spiegelinstanz als operative Oberflaeche.

3. **ExecComplex nur als erste Instanz, nicht als versteckte Semantik.** Die algebraischen Seeds werden definitorisch generisch ueber einer kleinen Ring-/Sternringoberflaeche formuliert; `ExecComplex` ist die erste konkrete Produktivinstanz, aber nicht die einzige moegliche spaetere Instanz.

4. **Bridge statt Zweitumbau.** Die spaetere Einbettung nach **C** wird von Anfang an als eigener strukturtreuer Brueckenbaustein geplant; spaetere spektrale, thermische oder analytische Pfeiler sollen diese Bruecke konsumieren koennen, ohne die A-Wurzel erneut umzuschreiben.

Vorbereitungsphasen V0–V5

Phase	Betroffene Module	Muss am Phasenende wahr sein
V0	`Foundation/MatrixNorms`, `Foundation/HermitianStructure`, `Foundation/FiniteHilbert`, `Foundation/MatrixAlgebra`, `Foundation/Notation`, `Foundation/BuildAll`, Repo-Audit	die Lean-/mathlib-Problematik der oeffentlichen komplexwertigen Seed-Oberflaeche ist explizit dokumentiert; es ist festgehalten, welche Teile nur algebraisch/exekutiv gebraucht werden und welche spaetere analytische Consumer bleiben; zugleich ist der algebraische Dualstrang als `ExecComplex`-Produktivpfad plus **C**-Spiegelpfad ausdruecklich als parallele Architektur festgehalten; jeder oeffentliche `noncomputable`-Rest in der algebraischen A-Wurzel ist als Planverstoß klassifiziert, sofern er nicht in die Whitelist (lokaler Solver, IDEAL-Transport, genuine Limesselektion oder spaetere **C**-Spiegelinstanz) faellt
V1	`Foundation/ExecComplex`, optional `Foundation/ExecComplexLemmas`	ein eigener ausfuehrbarer Datentyp `ExecComplex` mit expliziten Komponenten (zunaechst rational), `Zero`/`One`/`Add`/`Neg`/`Sub`/`Mul`/`Star`, mindestens `CommRing`- und `StarRing`-Struktur sowie elementaren Rechen- und Injektivitaetssaetzen steht; explizit festgehalten ist dabei, warum $\mathbf{Q}^2$ fuer den Generatorpfad zunaechst reicht und wo die Grenze liegt: diskrete Laplace-/Matrixoperationen laufen algebraisch/exekutiv, waehrend die sichtbare thermische Achse β im produktiven A-Pfad vorerst rationaler Provenienzparameter bleibt und ihre **R**-Lesart erst spaeter ueber die Bruecke erfolgt; `ExecComplex` ist oeffentlich computable und benoetigt keinen Rueckgriff auf die analytische **C**-Schicht
V2	`Foundation/MatrixNorms`, `Foundation/HermitianStructure`, `Foundation/FiniteHilbert`, `Foundation/MatrixAlgebra`	die algebraischen Seeds sind definitorisch generisch ueber einer kleinen Sternring-/Ringoberflaeche formuliert: Adjungierte/Hermitizitaet, finiter Zustandsraum, sesquilineare Seed-Oberflaeche, Matrixwirkung, $*$-Algebra, Kommutator und Antikommutator stehen *ohne* `RCLike`, `NormedField`, `InnerProductSpace` oder andere analytische Klassen als oeffentliche Voraussetzung; zugleich ist `MatrixNorms` als erster expliziter Dualstrang-Consumer so geschaerft, dass Frobenius-Quadrat, Positivitaets- und Shift-Oberflaeche auf der `ExecComplex`-Seite computable liegen, waehrend analytische Normsaetze nur im getrennten **C**-Spiegelstrang verbleiben; `ExecComplex` ist erste konkrete Instanz dieser allgemeinen Schicht
V3	`Foundation/MatrixNorms`, `Foundation/Notation`, `Foundation/Interfaces`, `Foundation/BuildAll`	es gibt eine saubere ausfuehrbare Spezialisierung fuer den Generatorpfad (z.B. `ExecMat`, `ExecState`); die Notation verdeckt weder den Koeffiziententyp noch abhaengige Transporte; `Interfaces` konsumiert die algebraische Wurzel nur ueber explizite Vertragsobjekte und fuehrt keinen zweiten semantischen Pfad neben die generische Seed-Oberflaeche ein; `MatrixNorms` exportiert seine operative Shift- und Vergleichsseite ebenfalls nur als `ExecComplex`-Produktivoberflaeche

Phase	Betroffene Module	Muss am Phasenende wahr sein
V4	`Foundation/ExecComplexBridge`, `Foundation/MatrixNorms`, ggf. kleine lokale Hilfssaetze in `HermitianStructure/FiniteHilbert/MatrixAlgebra`	eine spaetere Erweiterung nach **C** ist strukturell vorbereitet: es gibt einen strukturtreuen Morphismus `ExecComplex` →** **C** bzw. eine aequivalente explizite Abbildung, und **Gate-Bedingung dieser Phase ist ihre Injektivitaet**; nur so bleiben Matrixeintraege, Operatoren und spaetere Spektralaussagen sauber transportierbar. Dazu liegen Vertraeglichkeitssaetze fuer `star`, Matrixabbildung, Vektorabbildung, Adjungierte, inneres Produkt, Kommutator und die von `MatrixNorms` exportierten Frobenius-/Shift-Groessen vor; nur dieses Brueckenmodul darf die starke **C**-Schicht direkt importieren, und jede **C**-seitige Spiegelinstanz ist auditierbar als paralleler Nicht-Produktivstrang zu kennzeichnen
V5	alle in V0–V4 betroffenen Module, `StrengtheningPhase`-Audit, Phase S1	der Vorbereitungsblock ist in S1 reintegriert: die oeffentliche algebraische A-Wurzel ist wieder computable, `MatrixNorms` ist als erster expliziter Dualstrang-Consumer legal umgestellt, die spaetere Erweiterbarkeit nach **C** ist proof-carrying vorbereitet, die Audits greifen auf die neue Koeffizienten- und Brueckendisziplin, und Phase S1 darf erst dann als geschlossen gelten, wenn `MatrixNorms/HermitianStructure/FiniteHilbert/MatrixAlgebra` diese Disziplin sichtbar konsumieren

Folge fuer die eigentlichen Strengthening-Phasen

Der Vorbereitungsblock aendert die Ziele des Generator-Kernblocks S2–S14 nicht, verschiebt aber die Schliessungsbedingung von S1: S1 ist erst dann fachlich grun, wenn die computable Referenzfamilie *und* die oeffentliche algebraische A-Wurzel gemeinsam legal geschlossen sind. Zugleich reicht der bisherige Horizont S0–S14 nach der Scalar-Emergence-Note nicht mehr aus: oberhalb des Generator-Kernblocks ist ein expliziter Scalar-Emergence-Block S15–S20 einzuziehen. Wo der algebraische Vorbereitungsblock geoeffnet wird, ist er daher weiterhin als S1-Vorstufe bzw. S1a–S1e zu lesen, nicht als spaeter nachziehbarer Komfortblock.

Mit nun gruenem V0-Stand Die Vorbereitungsphase V0 ist auf dem aktiven Pfad geschlossen: `Foundation/HermitianStructure`, `Foundation/FiniteHilbert` und `Foundation/MatrixAlgebra` sind nun als neue algebraische Seed-Module im aktiven Build angelegt, `Foundation/Notation` und `Foundation/BuildAll` sind auf diese Vorstufe fortgeschrieben, und die Importkante nach `Foundation/Interfaces` bleibt dabei konsumdiszipliniert, also ohne die neue Algebra-Schicht implizit in jeden spaeteren Consumer einzuschmuggeln.

Inhaltlich schliesst V0 damit genau die beabsichtigte Vorbereitungsstufe und nicht mehr: Die algebraische A-Wurzel ist jetzt explizit als generische Seed-Oberflaeche mit spaeterem `ExecComplex`-Produktivpfad und getrenntem **C**-Spiegelpfad markiert, aber weder `ExecComplex` selbst noch die legale Umstellung von `MatrixNorms` auf den operativen Dualstrang noch die injektive Bruecke ExecComplex → **C** sind dadurch schon geschlossen. Folglich bleiben V1–V5 echte Restphasen; fuer S1 bedeutet der gruene V0-Stand deshalb eine geoeffnete und auditierbar vorbereitete algebraische Wurzel, aber noch nicht ihre endgueltige computable Closure.

Mit nun gruenem V1-Stand Die Vorbereitungsphase V1 ist auf dem aktiven Pfad geschlossen: `Foundation/ExecComplex` ist als eigener ausfuehrbarer Koeffiziententyp im aktiven Build angelegt, `Foundation/ExecComplexLemmas` stellt die elementaren Rechen-, Komponenten- und Injektivitaetssaetze der neuen Wurzel explizit bereit, und `Foundation/Notation` sowie `Foundation/BuildAll` konsumieren diese Vorstufe sichtbar, ohne dadurch bereits die spaeteren analytischen **C**-Consumer in die oeffentliche Produktivschicht hineinzuziehen. Zugleich ist die Importkante von `ExecComplex` bereinigt: der breite Komfortimport `Mathlib` ist entfernt, und der neue Koeffiziententyp haengt nur noch an der fuer `CommRing.ofMinimalAxioms`, rationale Koeffizienten, Sternstruktur sowie `ext/ring` tatsaechlich benoetigten Mathlib-Oberflaeche.

Inhaltlich schliesst V1 damit genau die beabsichtigte computable Wurzelstufe und nicht mehr: Der operative Produktivstrang besitzt nun mit `ExecComplex` eine explizite, oeffentlich computable

algebraische Basis mit `Zero/One/Add/Neg/Sub/Mul/Star`, mindestens `CommRing`- und `StarRing`-Struktur sowie den benoetigten elementaren Komponenten- und Rechenfakten. Dabei bleibt die rationale $\mathbf{Q}^2$-Darstellung bewusst ein Startpunkt des Generatorpfads und kein verdeckter semantischer Endzustand: Weder ist `MatrixNorms` damit schon legal auf den operativen Dualstrang umgestellt noch ist die injektive Bruecke ExecComplex $\to$ **C** bereits geschlossen. Folglich bleiben V2–V5 echte Restphasen; fuer S1 bedeutet der gruene V1-Stand deshalb eine nun oeffentlich computable algebraische A-Wurzel, aber noch nicht die vollstaendige Re-Integration des Vorbereitungsblocks in die Strengthening-Hauptkette.

Mit nun gruenem V2-Stand Die Vorbereitungsphase V2 ist auf dem aktiven Pfad geschlossen: `Foundation/MatrixNorms`, `Foundation/HermitianStructure`, `Foundation/FiniteHilbert` und `Foundation/Ma` konsumieren die algebraische A-Wurzel nun sichtbar ueber eine kleine generische Sternring-/Ringoberflaeche statt ueber oeffentliche analytische Klassen, und die Importdisziplin der V2-Module ist auf die tatsaechlich benoetigten Mathlib-Bausteine zurueckgenommen. Insbesondere ist `MatrixNorms` als erster expliziter Dualstrang-Consumer legal geschaerft: die operative Frobenius-, Positivitaets- und Shift-Oberflaeche liegt auf der `ExecComplex`-Seite computable, waehrend der **C**-Spiegelstrang getrennt und auditierbar bleibt.

Inhaltlich schliesst V2 damit genau die beabsichtigte generische Algebra- und Dualstrangstufe und nicht mehr: Die oeffentliche algebraische A-Wurzel ist nun nicht nur computable vorhanden, sondern in ihren ersten Seed-Consumern auch sichtbar legal konsumiert. Noch offen bleiben jedoch die saubere ausfuehrbare Spezialisierung des Generatorpfads ueber explizite Vertragsobjekte in `Foundation/Interfaces`, die proof-carrying Bruecke ExecComplex $\to$ **C** sowie die vollstaendige Re-Integration des Vorbereitungsblocks in S1. Folglich bleiben V3–V5 echte Restphasen; fuer S1 bedeutet der gruene V2-Stand deshalb eine nun sichtbar legalisierte algebraische A-Wurzel, aber noch nicht den vollstaendigen Abschluss des Vorbereitungsblocks. Fuer den Generator-Kernblock oberhalb von V2 ergibt sich daraus keine Umplanung; Die Reihenfolge V3 $\to$ V4 $\to$ V5 bleibt verbindlich, und insbesondere darf ein etwaiger analytischer **C**-Spiegelpfad bis zur geschlossenen Brueckenphase V4 nicht als operative Produktionsoberflaeche einspringen.

Mit nun gruenem V3-Stand Die Vorbereitungsphase V3 ist auf dem aktiven Pfad geschlossen: `Foundation/MatrixNorms`, `Foundation/Notation`, `Foundation/Interfaces` und `Foundation/BuildAll` tragen nun eine sichtbare ausfuehrbare Spezialisierung des algebraischen Produktivstrangs. Insbesondere exponiert `Interfaces` die operative A-Wurzel ueber explizite Vertragsobjekte und generische Netzhuellen, statt einen zweiten semantischen Pfad neben die bereits in V2 legalisierte Seed-Oberflaeche zu setzen; zugleich bleibt `MatrixNorms` auch auf dieser Spezialisierungsstufe strikt auf die computable `ExecComplex`-Produktivseite beschraenkt. Die in der Implementierung nachgezogene `SeedScalar`-Instanzpropagation fuer `ExecComplex` macht dabei explizit sichtbar, dass V3 nicht nur Alias-/Notationskosmetik schliesst, sondern die algebraische Wurzel als konsumierbare operative Instanz wirklich bis in die Vertrags- und Spezialoberflaeche hinein traegt.

Inhaltlich schliesst V3 damit genau die beabsichtigte ausfuehrbare Spezialisierungs- und Vertragsstufe und nicht mehr: Der Generatorpfad besitzt nun eine lesbare operative Oberflaeche ueber `ExecMat`, `ExecState` und verwandte Spezialisierungen, ohne dabei den Koeffizententyp oder abhaengige Transporte semantisch zu verdecken, und `Interfaces` bleibt weiterhin strikt Consumer der algebraischen Wurzel statt selbst neue Grundsemantik zu erfinden. Noch offen bleiben jedoch die proof-carrying Bruecke ExecComplex $\to$ **C** sowie die vollstaendige Re-Integration des Vorbereitungsblocks in S1. Folglich bleiben V4–V5 echte Restphasen; fuer S1 bedeutet der gruene V3-Stand deshalb eine nun operativ konsumierbare und vertragsseitig sichtbare algebraische A-Wurzel, aber noch nicht den vollstaendigen Abschluss des Vorbereitungsblocks. Fuer den Generator-Kernblock oberhalb von V3 ergibt sich daraus keine Umplanung; Die Reihenfolge V4 $\to$ V5 bleibt verbindlich, und insbesondere darf der analytische **C**-Spiegelpfad bis zur geschlossenen Brueckenphase V4 weiterhin nicht als operative Produktionsoberflaeche einspringen.

Mit nun gruenem V4-Stand Die Vorbereitungsphase V4 ist auf dem aktiven Pfad geschlossen: `Foundation/ExecComplexBridge` ist als eigene proof-carrying Brueckenstufe im aktiven Build angelegt, `Foundation/MatrixNorms` konsumiert den analytischen **C**-Spiegelpfad nicht mehr implizit, sondern nur noch ueber die explizite Brueckendisziplin, und die Implementierung traegt die benoetigten Transport- und Instanzsaetze fuer `star`, Matrix- und Vektorabbildung, Adjungierte, inneres Produkt, Kommutator sowie die von `MatrixNorms` exportierten Frobenius-/Shift-Groessen sichtbar im neuen Brueckenmodul. Die im Build erzwungene `DecidableEq`-Propagationsdisziplin macht dabei zusaetzlich auditierbar, dass die finite Transportoberflaeche ohne vermeidbare `classical`-Abkuerzungen geschlossen ist.

Inhaltlich schliesst V4 damit genau die beabsichtigte proof-carrying Brueckenstufe und nicht mehr: Die spaetere Erweiterung nach **C** ist nun strukturell ueber einen injektiven Morphismus ExecComplex → **C** vorbereitet, und der analytische Spiegelstrang ist als paralleler Nicht-Produktivstrang explizit von der operativen `ExecComplex`-Produktivseite getrennt. Noch offen bleibt jedoch die vollstaendige Re-Integration des Vorbereitungsblocks in Phase S1. Folglich bleibt V5 die einzige Restphase dieses Blocks; fuer S1 bedeutet der gruene V4-Stand deshalb eine nun proof-carrying vorbereitete algebraische A-Wurzel mit geschlossener Bruecke nach **C**, aber noch nicht den endgueltigen Abschluss des Vorbereitungsblocks. Fuer den Generator-Kernblock oberhalb von V4 ergibt sich daraus weiterhin keine Umplanung; Es bleibt bei V5 als reinem Re-Integrations- und Audit-Schluss, und die Ziele von S2–S14 bleiben unveraendert.

Mit nun gruenem V5-Stand Die Vorbereitungsphase V5 ist auf dem aktiven Pfad geschlossen: Die Re-Integrations- und Audit-Stufe des algebraischen Vorbereitungsblocks ist nun explizit in Phase S1 eingezogen, die grep-pruefbare Koeffizienten- und Brueckendisziplin bleibt auf dem aktiven Pillar-A-Pfad erhalten, und die kleine offene Notationsunschaerfe an der Handoff-Grenze ist bereinigt. Insbesondere spiegelt `Handoff/Notation` die aktive oeffentliche Handoff-Oberflaeche nun wieder vollstaendig fuer `SectorExport`, `Step1StrongCore`, `Step1MathData`, `ABHandoffStrong` und `PillarAStep1Closed`; zugleich ist explizit festgehalten, dass verbleibende lokale Notation in Ursprungsmodulen nur dort zulaessig ist, wo eine Konsumption der spaeteren Wrapper wegen Importzyklizitaet nicht moeglich waere und dadurch kein zweiter oeffentlicher Semantikpfad entsteht.

Inhaltlich schliesst V5 damit genau den beabsichtigten Re-Integrations- und Audit-Schluss und nicht mehr: Die oeffentliche algebraische A-Wurzel ist nun wieder computable, `MatrixNorms` bleibt als erster expliziter Dualstrang-Consumer legal auf der operativen `ExecComplex`-Seite verankert, die spaetere Erweiterbarkeit nach **C** ist proof-carrying vorbereitet, und die sichtbare Konsumption dieser Disziplin durch `MatrixNorms`/`HermitianStructure`/`FiniteHilbert`/`MatrixAlgebra` ist zusammen mit der bereinigten Handoff-Notationsspiegelung auditiert. Fuer den Generator-Kernblock oberhalb von V5 ergibt sich weiterhin keine Umplanung; Der Generator-Kernblock S2–S14 bleibt in seinen Nahzielen unveraendert; der Gesamtpfad wird jedoch um einen expliziten Scalar-Emergence-Block S15–S20 erweitert. Global schliesst dies Phase S1 jedoch weiterhin nicht allein, weil die computable konkrete Referenzfamilie als zweites S1-Zahnrad separat gruen stehen muss.

Mit nun gruenem S4-Stand Die Strengthening-Phase S4 ist auf dem aktiven Pfad in ihrem Kern geschlossen: `ToC/Concrete` exponiert nun mit `referenceToC` und `variationToC` zwei legale Eintrittsobjekte derselben `ToCStrong`-Oberflaeche, und `Finite/CutSpec`, `Finite/RegionCore`, `Finite/BoundaryPorts`, `Finite/Approximant` sowie flankierend `Finite/Selection` konsumieren diese beiden Einstiege sichtbar ueber denselben generischen endlichen Box-Pfad. Damit ist der in S4 geforderte harte Typcheck nicht mehr nur proseartig, sondern durch die beiden realen Familienlaeufe als getypte Konsumption des gemeinsamen Downstream-Builders erbracht; insbesondere bleibt der Referenzlauf kein Smoke-Test-Sonderpfad neben der Variationsfamilie. Ebenso ist die ab S3 geforderte Notationsdisziplin nun materiell eingeloest: die freigegebene Referenzoberflaeche bleibt nicht bloss global gespiegelt, sondern wird im realen Finite-Downstream – insbesondere bereits in `Finite/CutSpec` – operativ sichtbar konsumiert.

Inhaltlich schliesst S4 damit genau die beabsichtigte gemeinsame Eintritts- und Finite-Unifikationsstufe und nicht mehr: Referenzfall und bereits legalisierte Variationsfamilie laufen nun ab dem ersten realen `ToCStrong`-/Approximantenkern durch denselben kanonischen Box-Pfad, ohne neue `noncomputable`-Produktionsdefinitionen, `classical`-Hilfen, `native_decide`-Ersatzpfade oder pauschale `@[simp]`-Flaechen in die betroffenen S4-Module einzuziehen. Zwei kleine Restpunkte bleiben dabei explizit als Audit- und nicht als Architekturblocker markiert: erstens ist ein derzeit unnoetiger Import von `Foundation/Determinism` in `ToC/Concrete` unter strenger Consumer-Disziplin entweder noch zu entfernen oder spaeter sichtbar zu konsumieren; zweitens ist die Variationsseite notationsmaessig noch nicht so symmetrisch nach `ToC/Notation` gespiegelt wie die Referenzseite. Beides blockiert den S4-Schluss nicht, ist jedoch spaetestens bis zur allgemeinen Notationsbereinigung in S12 sauber nachzuziehen. Fuer den Generator-Kernblock oberhalb von S4 ergibt sich daraus keine Umplanung: Der naechste echte Motorblock bleibt S5 mit `Finite/DirichletLaplacian` und `DtN/DtN`; dort sind `DirichletWeight` an eine explizite Policy-/Builderoberflaeche zu binden und `InteriorInverse` als zu internalisierende Solver-/Eliminationslogik weiter voranzutreiben.

Mit nun gruenem S5-Stand Die Strengthening-Phase S5 ist auf dem aktiven Pfad in ihrem Kern geschlossen: `Finite/DirichletLaplacian` bindet `DirichletWeight` nun nicht mehr als freien Strong-Eingang, sondern derived-only ueber `ApproximantStrong` und `WeightPolicy`; damit bleiben Referenzlauf und Variationslauf auch auf der ersten echten Operatorstufe sichtbar im selben Produktivpfad. Parallel dazu exponiert `DtN/DtN` keinen oeffentlichen freien `InteriorInverse`-Strong-Feldrest mehr, sondern eine explizite Solver-/Eliminationsoberflaeche mit Eliminationsmodus, Randoperator, Innen-Solve und Korrektheitssaetzen. Damit ist die Inversionsseite im aktiven Pfad nicht mehr als lose externe Eingabe formuliert, sondern als proof-carrying Eliminationsschritt derselben legalen Provenienz. Die oeffentlichen Familienbuilder laufen folglich ohne explizites Inversenargument durch denselben Referenz-/Variationspfad weiter; die lesbare S5-Notation fuer Gewichts- und Eliminationsseite ist freigegeben, ohne die Provenienz der Operatorstufe semantisch zu verdecken.

Inhaltlich schliesst S5 damit genau die beabsichtigte erste Generatorengstellen-Schaerfung und nicht mehr: Weder wird bereits die Shift-/Stabilisierungsseite von `DtNStabilized` vorweggenommen noch die spaetere Closure-Semantik in die binaere DtN-Stufe zurueckverlagert. Im betroffenen S5-Delta kommen keine neuen pauschalen `@[simp]`-Flaechen, keine neuen oeffentlichen `noncomputable`-Produktionsdefinitionen und keine neuen `classical`-Hilfen hinzu; die tragende Importkante bleibt strikt `Approximant` → `DirichletLaplacian` → `DtN`. Fuer den Generator-Kernblock oberhalb von S5 ergibt sich daraus weiterhin keine Umplanung: Der naechste echte Motorblock bleibt S6 mit `DtN/DtNStabilized`, `Closure/ParameterClosure` und `Closure/RegularizationClosure`; dort ist nun die Shift-, Stabilisations- und Closure-Seite auf Basis der in S5 legalisierten Operator- und Eliminationsoberflaeche weiter zu internalisieren.

13 Strengthening-Modulliste S0–S20

Für flankierende Diagnose- und Limesmodule wird die IDEAL/REAL-Trennung im Feld Muss lokal schliessenëxplizit markiert, damit die Trennungsregel nicht nur in den Phasengates, sondern bereits in der Modulliste pruefbar bleibt.

Nr.	Modul / Bereich	Aktion	Muss lokal schließen	Muss wie verzahnen
1	`Foundation/SubstrateClass`	ändern	die bereits vorhandene typklassenparametrische Universalität explizit als produktive Generatorbasis ausweisen; kein zweiter Meta-Vertrag neben `SubstrateClass`	bleibt primitive Wurzel für alle späteren Consumer und für jede legale Instanziierung des Produktivpfads

Nr.	Modul / Bereich	Aktion	Muss lokal schließen	Muss wie verzahnen
2	`Foundation/MatrixNorms`, `Foundation/Interfaces`, `Foundation/Determinism`	ändern	Matrix-, Norm-, Spur- und Instanzgrundlage so schärfen, dass endliche Operator-, Zustands-, Netz- und Diagnosepfade nicht auf informelle Hilfslogik ausweichen; `Foundation/Interfaces` trägt dabei ausdrücklich generische Netz-, Prägarben-, Koprägarben- und Vertragsobjekte wie `RegionNet`, `PreNet`, `LocalAlgebraNet`, `StateNet`, `ChannelNet` sowie einen A-seitig benannten `StarAlgebraContract`; deterministische Zusätze bleiben flankierende Verfeinerung	versorgt `Contract`, finite Operatorseeds, spätere Handoff-Instantiierungen und Audits, ohne eine versteckte stärkere Importvoraussetzung in P7–P27 einzuschmuggeln
3	`Foundation/HermitianStructure`	hinzufügen	Adjungierte, Hermitizität/Selbstadjungiertheit, unitäre Matrizen und die zugehörigen elementaren Rechenregeln über endlichen Matrixcarriern als *generische* Sternring-Wurzel bereitstellen; keine öffentliche Bindung an **C**-Analyseklassen	liefert die algebraische Vorstufe für Spektralzerlegung, $*$-Algebra, Zeitentwicklung und spätere GNS-/KMS-Pfade, ohne die ausführbare Generatorwurzel an die analytische Bibliothek zu ketten
4	`Foundation/FiniteHilbert`	hinzufügen	endliche Zustands- und Inner-Product-Seed-Oberfläche über demselben algebraischen Koeffizienten bereitstellen; Orthonormalbasis, Matrixwirkung und Spur-/Inner-Product-Zusammenhang derived-only und öffentlich computable, ohne vorzeitig lineartopologische Klassen einzuführen	macht Spektralzerlegung, Zustandsraum und spätere Darstellungsseeds als gemeinsame Consumer derselben Hilbertraumwurzel lesbar und hält die spätere ExecComplex $\to$ **C**-Erweiterung offen
5	`Foundation/MatrixAlgebra`	hinzufügen	endliche Matrix-$*$-Algebra über derselben generischen Sternring-Wurzel samt Kommutator, Antikommutator und elementaren Algebra-/Lie-Sätzen bereitstellen; konkrete Produktivinstanz ist `ExecComplex`, nicht blind **C**	verhindert B-/C-seitigen Nachbau der algebraischen Sprache aus bloßen Matrixoperationen und hält eine strukturtreue Brücke in die komplexanalytische Welt explizit vorbereitet
6	`Foundation/ConcreteSubstrate`	ändern	bestehende konkrete Referenzfamilie des regulären konkreten IDEAL-ToC so reparieren, dass die `SubstrateClass`-Instanz ohne `sorry`s und ohne vermeidbare `noncomputable`-Kontamination steht	liefert den ersten robusten Referenzlauf des späteren Generators
7	`Foundation/LevelVariableSubstrate`	hinzufügen	zweite legale Substratfamilie mit levelabhängigem Branching als echter Variationsfall; Determinismus darf hier, falls vorhanden, nur Zusatz und nicht Kernannahme sein	liefert den ersten belastbaren Nicht-Referenzfall für Universalisierungs- und Diskriminanzsätze
8	`Foundation/WeightPolicy`, `Foundation/RegularizationPolicy`	hinzufügen	Gewichte, numerisch-kanonische Regularisierung und die sichtbare thermische Achse β als explizite A-seitige Variations- und Provenienzparameter modellieren; keine versteckte Policy-Logik in `DirichletLaplacian`, `DtN` oder späteren Seeds; Legacy-`Policy`-Felder wie `rStar`, `alpha` oder `alphaDenom` gelten dabei nicht als primitive neue Motorachsen, sondern im Referenzfall als numerische bzw. derived-only Hilfsgroessen, solange keine spaetere mathematische Notwendigkeit fuer eine eigenstaendige Variationsachse nachgewiesen ist	zieht `DirichletWeight` und die Herkunft von ϵ, β aus der Mittelstufe an die Motorseite des Generators, ohne ihre spaetere pfeileruebergreifende Schliessung zu behaupten
9	`Foundation/SubstrateAnalysis`	hinzufügen	explizit klassifizieren, welche Resultate wirklich nur `SubstrateClass` benötigen und welche stärkere Zusatzannahmen wie Determinismus oder Uniformität verbrauchen	macht Universalität und echte Variationsaussagen formal statt nur proseartig sichtbar

Nr.	Modul / Bereich	Aktion	Muss lokal schließen	Muss wie verzahnen
10	`ToC/Contract`, `ToC/Addressing`, `ToC/IdealAddressEquiv`, `ToC/ConcreteIdeal`	ändern/hinzufügen	den konkreten Referenzfall des IDEAL-ToC als Familie schließen, sauber an Contract/Addressing/Equiv rückbinden und die zugehörige Referenz-/Aliasoberfläche in `ToC/Notation` sowie `PillarA/Notation` freigeben; bloße Spiegelung ohne spätere reale Konsumption genügt dabei noch nicht als Endzustand des Generators	liefert den konkreten Idealstartpunkt, aus dem `ToCStrong` produktiv hervorgeht, und markiert die Notationsoberfläche, die ab S4 operativ einzulösen ist
11	`ToC/Concrete`	ändern	den ersten realen `ToCStrong`-Kern so formulieren, dass Referenzfall und Familieninstanz denselben Downstream-Pfad öffnen; die in S3 freigegebene Referenznotation darf hier nicht bloß mitlaufen, sondern muss in den realen Downstream-Consumern sichtbar konsumiert werden	versorgt den Finite-Pfad ohne Smoke-Test-Sonderroute und zieht die S3-Notation in den operativen Pfad hinein
12	`Finite/CutSpec`, `Finite/RegionCore`, `Finite/BoundaryPorts`, `Finite/Approximant`, optional `Finite/Selection`	ändern	den endlichen Box-Pfad vereinheitlichen, sodass beide Familienfälle ab hier wirklich dieselben Consumer bedienen und als identischer Buildpfad überprüfbar bleiben; zugleich müssen alle exponierten Carrier/Prädikate die nötigen `Fintype`-/`Decidable`-Instanzen mitliefern, und die aus S3 kommende Referenznotation darf nicht wieder auf reine Alias-Prosa zurückfallen	überführt alle legalen Einstiege in denselben kanonischen endlichen Box-Inhalt und macht die S3-Spiegelung im Finite-Pfad operativ sichtbar
13	`Finite/DirichletLaplacian`	ändern	den noch freien `DirichletWeight`-Eingang explizit als legitimen A-seitigen Wurzelinput über `WeightPolicy` modellieren; Referenzfälle dürfen zusätzliche kanonische Default-Gewichte tragen, die universelle Theorie aber nicht	bleibt erster operatorischer Consumer des produktiven Generatorpfads und kennt Gewichte nur über die freigegebene Policy-Verzahnung
14	`DtN/DtN`	ändern	die aktuelle nackte `InteriorInverse`-Feldform nicht als Wurzelinput, sondern als zu internalisierende Solver-/Eliminationslogik behandeln; Ziel ist eine algorithmische Ableitung mit öffentlicher Solve-/Korrektheitsoberfläche, wobei ein eventuell verbleibender `noncomputable`-Zeuge nur intern und lokal sichtbar bleiben darf	hält den binären DtN-Kern als echten Folgeconsumer von `DirichletLaplacianStrong` und verhindert eine lose Inversenschraube im Generatorpfad
15	`DtN/DtNStabilized`	ändern	freie `epsilon`-Seite so weit legal möglich in eine kanonische computable Shift-/Regularisierungsprovenienz überführen	bleibt einziger legaler Stabilisierungsconsumer für Closure- und Coupling-Pfade
16	`Closure/ParameterClosure`, `Closure/RegularizationClosure`	ändern	Closure-Seite so verschärfen, dass primitive Restparameter nicht unkommentiert weitergereicht werden und die aus P13 kommende Provenienz explizit bleibt	exportiert eine operative Closure-Oberfläche für spätere Handoffs und Vergleiche
17	`Coupling/GeneralizedDtN`	ändern	raw/stabilized-Dualität, Restriktionsdaten und operatorische Provenienz familiesicher schärfen; keine Reparatur durch spätere Exportmodule	koppelt Operatorpfad und Sektorseite explizit als legalen Consumerblock
18	`Coupling/MultiSchur`	ändern	die aktuelle nackte `interfaceInverse`-Feldform nicht als Wurzelinput, sondern als innerhalb von P22 zu internalisierende Restriktions-/Solver- bzw. Reduktionslogik behandeln; Exportziel sind Schur-/Glueing-/Korrektheitssätze statt einer losen Inversenmatrix	bleibt reiner Folgeconsumer von `GeneralizedDtNStrong` und hält die P22-Fachlogik von freien Schrauben frei

Nr.	Modul / Bereich	Aktion	Muss lokal schließen	Muss wie verzahnen
19	`Network/InfiniteCarrier`	ändern	stabile endliche Ausschnitte und gerichtete Übergänge so exportieren, dass Referenzfall und Variation auf derselben Trägeroberfläche vergleichbar bleiben; limesnahe Provenienz darf nicht hinter späteren Exportadaptern verschwinden	liefert die späte Trägeroberfläche für Handoff, Variation und spätere Limesdiagnostik
20	`Network/RegionNet`, `Network/SectorChannels`	unverändert prüfen	keine fachliche Neuformulierung im Strengthening; explizit festhalten, dass diese Supportmodule in Schritt 2 nur als bereits geschlossene legale Vorgänger konsumiert und nicht stillschweigend mitbearbeitet werden	verhindert Scheinarbeit im Supportpfad und hält die Phasenzuordnung prüfbar
21	`Network/SectorSysEnv`, `Network/RelativeEntropyFlow`, `Geometry/LCPMeasure`, `Geometry/Foliation`, `Geometry/SpacetimePaths`	ändern	IDEAL/REAL getrennt: ja. Flankierende Skalen-, Schicht-, Präfix-, Window- und Environment-Provenienz für spätere Spektraldimensions- sowie Symmetrie-/Cutoff-/Limesdiagnostik derived-only sichtbar halten und dabei die IDEAL-Seite strikt von der REAL-Seite trennen	liefert A-seitige Diagnosewerkzeuge für spätere Pfeiler, ohne den Generatorhauptpfad rückwärts zu verunreinigen
22	`Network/DirectedLimit`	hinzufügen	IDEAL/REAL getrennt: ja. Gerichtete Übergänge, Konvergenzbegriff und Limes-Eindeutigkeit für die Lesart $\mathrm{REAL}_\infty \to \mathrm{IDEAL}_\infty$ formalisieren, ohne bereits B- oder AQFT-Limessemantik vorwegzunehmen	macht limesseitige Aussagen aus `InfiniteCarrier` und Exportdaten formal statt nur proseartig
23	`Finite/SpectralDecomposition`, `Finite/SpectralDecompositionC`, `Finite/SpectralBridge`, sichtbarer Folgeblock `Finite/ExecSpectral/*`	hinzufügen	die endliche Spektralwurzel als expliziten Dualstrang aufbauen: `SpectralDecomposition` bleibt die einzige öffentliche operative `ExecComplex`-Oberfläche; `SpectralDecompositionC` trägt nur die analytische **C**-Spiegelung nach Whitelist (d); `SpectralBridge` theoremiert die injektive Transferlage; der sichtbare Folgeblock `ExecSpectral/*` bereitet einen moeglichst allgemeinen computablen Hermitesch-Pfad mit Charakteristikum, Wurzelisolation, Eigenvektorkernen, Projektoren und Zertifikaten strukturell vor	liefert B-seitig benötigte spektrale Theoremsamen aus A-generierten Operatoren, ohne AQFT-Semantik nach vorn zu ziehen, und hält zugleich den operativen Produktivpfad von einem stillschweigenden zweiten **C**-Generator getrennt
24	`Finite/StateSpace`	hinzufügen	positiven Kegel, Zustandsraum, Spurform und Normierungsoberfläche über endlichen Matrixcarriern derived-only und möglichst computable bereitstellen	verhindert, dass spätere Pfeiler den endlichen Zustandsraum aus `MatrixNorms` und eigener Hilfslogik neu aufbauen
25	`Finite/MatrixExponential`, `Finite/GibbsStateSeed`, `Finite/DynamicsAdapter`	hinzufügen	Matrixexponentialfunktion, Partitionsfunktion, diagonalen Gibbs-Zustand mit Positivitaet und Normierung sowie den Heisenberg-Adapter $\alpha_t(A) = U(t)AU(t)^\dagger$ als `StarAlgDynamics`-Vorstufe über endlichen A-seitigen Operatoren bereitstellen, ohne bereits KMS-/AQFT-Semantik zu behaupten	liefert thermische und dynamische Seeds für spätere B-Pfade direkt aus A-generierten Operatoren und verhindert B-seitigen Nachbau roher Matrixarithmetik
26	`Finite/UnitaryEvolution`	hinzufügen	unitäre Einparametergruppe $t \mapsto e^{-itH}$ samt Gruppen-, Unitaritäts- und elementaren Heisenberg-Sätzen über der endlichen Matrix-$*$-Algebra bereitstellen; keine volle Dynamikaxiomatik	liefert die dynamische Wurzel für spätere Heisenberg-, Modular- und OQS-Verbraucher, ohne B/C-Fachlogik vorwegzunehmen

Nr.	Modul / Bereich	Aktion	Muss lokal schließen	Muss wie verzahnen
27	`Finite/ChannelSeed`	hinzufügen	Kraus-/Choi-/CPTP-Vorstufe und gegebenenfalls dissipative Halbgruppen-Seeds soweit bereitstellen, wie sie rein algebraischer Consumer der endlichen A-Matrixseite bleiben; keine Vorwegnahme von C-Fachlogik	verhindert verbotenen C-seitigen Nachbau der algebraischen Kanalwurzel und hält die kanalische Vorstufe dennoch strikt unterhalb echter OQS-Fachlogik
28	`Handoff/SectorExport`, `Handoff/Step1StrongCore`	ändern	Export- und Verdichtungsstufe um echte Generator-, algebraische, spektrale, zustandsräumliche, thermische, unitäre und Diagnoseprovenienz erweitern; insbesondere IDEAL-seitige Grenz-, Träger- und Äquivalenzdaten getrennt von REAL-seitigen Cutoff-, Komplement-, Übergangs- und Limesdaten zielneutral nach außen stellen	projiziert nur geschlossene reife A-Daten nach außen und macht spätere Pfeiler unabhängig von informellen Rekonstruktionen oder algebraischem Nachbau
29	`Handoff/Step1MathData`, `Handoff/ABHandoffStrong`, `Handoff/PillarAStep1Closed`	ändern	Endstufe so verschärfen, dass Referenzlauf und Variationslauf denselben proof-carrying Handoff schließen; `ABHandoffStrong` bleibt nur ausgehender Adapter und wird weder von A selbst konsumiert noch in B durch A-Nachbau ersetzt	bleibt einzig legitimer Ausgang des produktiven Pillar-A-Pfads und schließt Zirkularität zwischen Kern, Export und Adapter aus
30	`PillarA/Generator`	hinzufügen	expliziter Root-to-Handoff-Builder über dem Dreiklang *(Substrat, WeightPolicy, Cutoff)*; zusätzlich ein explizites Computability-Budget der Zahnräder festhalten; öffentliche Exportoberfläche soll vollständig computable sein, soweit nicht eine explizit whitelistete Restgrenze greift	macht den produktiven Smoke-Test zum echten Generatorlauf desselben aktiven Pfads und verhindert schleichende `noncomputable`-Erosion
31	`PillarA/VariationAnalysis`	hinzufügen	Vergleichsoberfläche für substratnahe, policy-, regularisierungs- und beta-nahe, cutoffnahe und genuinely universelle Invarianten; Theoreme sind explizit nach Universalitätstyp zu klassifizieren	erlaubt Aussagen darüber, welche Resultate universell, substratabhängig, policyabhängig, ϵ-/β-abhängig oder cutoffabhängig sind
32	`PillarA/Diagnostics/SpectralDimension`	hinzufügen	IDEAL/REAL getrennt: ja. Flankierende Diagnosevorstufe mit explizit getrennter IDEAL-Seite und REAL-Seite; keine Vorwegnahme eines physikalischen Spektraldimensionssatzes	bereitet spätere RG-, Backreaction- und Spektraldimensionspfade innerhalb von A vor, damit B und folgende Pfeiler keinen A-Code nachtragen müssen
33	`PillarA/Diagnostics/SymmetryCutoffLimit`	hinzufügen	IDEAL/REAL getrennt: ja. Flankierende Diagnosevorstufe mit explizit getrennter IDEAL-Seite und REAL-Seite; keine fertigen physikalischen Brechungssätze	stellt spätere Symmetrie-, Cutoff- und Limeswerkzeuge A-seitig bereit und hält ihre Herkunft strikt von späterer AQFT-/Modularsemantik getrennt
33a	`PillarA/Diagnostics/DualObserverSeed`, `PillarA/Diagnostics/InterfaceEntropyBound`, `PillarA/Diagnostics/SpectralGapSeed`, `PillarA/Diagnostics/SchurBlockSymmetrySeed`, `PillarA/Diagnostics/HorizonDefinition`, `PillarA/Diagnostics/CayleyDicksonSpectralShift`	hinzufügen	A-seitig derived-only eine duenne S10c-Diagnostikoberflaeche fuer duale Equivarianz der Seed-Kette, Interface-Entropiebound, kleinsten positiven DtN-Spektralabstand als Masse-/Horizont-Analogon, duale Symmetrie der Schur-Bloecke, Horizontkriterium $\lambda^{+}_{\min} \to 0$ sowie spektrale Verschiebung entlang der Cayley–Dickson-Fortsetzung bereitstellen; keine B/C/D/E-Imports, keine physikalische Casimir-, Feynman- oder Hurwitz-Schliessung	schliesst genau die algebraischen und spektralen Vorstufen, die spaetere Casimir-/Horizont-/Projektionslesarten in D nur noch konsumieren, aber nicht in spaeteren Pfeilern nachbauen duerfen
34	`Foundation/Notation`, `ToC/Notation`, `Finite/Notation`, `DtN/Notation`, `Closure/Notation`, `Coupling/Notation`, `Network/Notation`, `Geometry/Notation`, `Handoff/Notation`, `PillarA/Notation`	ändern	neue Lesbarkeitsoberflächen für Referenzfamilie, Variationsfamilien, `WeightPolicy`, Hermitian-/Hilbert-/Algebra-Seeds, Spektral-/Zustands-/Thermal-/Dynamikseeds, Generator, Directed-Limit und Diagnosemodule einziehen, ohne Parameter oder abhängige Transporte zu verdecken	hält den Strengthening-Pfad reviewbar und bleibt strikt scoped sowie semantisch neutral

Nr.	Modul / Bereich	Aktion	Muss lokal schließen	Muss wie verzahnen
35	`Foundation/BuildAll`, `ToC/BuildAll`, `Finite/BuildAll`, `DtN/BuildAll`, `Network/BuildAll`, `Geometry/BuildAll`, `PillarA/Structural/BuildAll`, `PillarA/ScalarEmergence/BuildAll`, `Handoff/BuildAll`, `PillarA/BuildAll`, `CNNA/BuildAll`	ändern	neue Module legal in den aktiven Build einhängen; ab S20 werden die A-internen Struktur- und Scalar-Emergence-Bloecke ueber `PillarA/BuildAll` integriert; keine zweite Schatten-Topologie und kein Rückgriff auf Archivpfade	macht den Strengthening-Pfad zum einzigen aktiven Produktions-, Diagnose- und Smoke-Test-Baum
36	`PillarA/ScalarEmergence/PreNumericSectorCore`	hinzufügen	eine explizite theorematische Extraktionsstufe bereitstellen, in der helle/interface/dunkle Carrier als prae-numerische Sektoroberflaeche formuliert werden; keine Zahlen-, Operator- oder Kopplungsannahmen in den oeffentlichen Saetzen dieser Stufe	macht auditierbar sichtbar, was an der Sektorseite bereits ohne Skalarwahl traegt, und liefert die begriffliche Wurzel fuer spaetere Scalar-Emergence-Seeds
37	`PillarA/ScalarEmergence/StatusLedger`	hinzufügen	alle Scalar-Emergence-Aussagen des Plans explizit nach Status klassifizieren: maschinengeprueft, externer Satz, strukturelle Beobachtung, offene Forschungsfrage oder Fernziel; zusaetzlich pro Stufe die jeweilige Versagensart knapp notieren (insbesondere **H**: Matrixalgebra noch assoziativ, Tensoring scheitert an Nichtkommutativitaet; **O**: bereits Matrixmultiplikation nicht mehr assoziativ), externe Anker wie Hurwitz, Solèr 1995, Barnum/Müller/Ududec 2014 und Renou et al. 2021 getrennt fuehren sowie Fernziele wie die Cayley-Dickson↔Generationen-Lesart mit Abhaengigkeiten und Literaturankern markieren; kein Vermischen von Generatorclosure und Forschungsbehauptung	wird von Diagnose, Handoff und spaeteren Pfeilern konsumiert, haelt die Forschungsgrenze selbst im Export auditierbar und macht die laenger werdende Hypothesenliste als echten Fortschrittsledger statt als lose Prosa sichtbar
38	`PillarA/ScalarEmergence/OneSectorRealSeed`	hinzufügen	eine explizite $\mathbf{R}_{\geq 0}$-artige Einzelsektor-Oberflaeche aus Zaehldaten, Gewichten, Positivitaet und Normierung vorbereiten, ohne bereits Kopplungs- oder Phaseninformation hineinzuschmuggeln	verbindet prae-numerische Sektorstruktur mit StateSpace-/Thermalseeds und macht die ontische Schicht *vor* der komplexen Kopplung planlesbar
39	`PillarA/ScalarEmergence/ComplexCouplingSeed`	hinzufügen	die vier Bedingungen Interferenz, Reversibilitaet, Positivitaet und Kompositionalitaet als explizite Constraint-/Hypothesenoberflaeche ueber bright-dark-Kopplungen formulieren; diese Viererliste ist im Plan zunaechst als Mindestliste zu lesen, nicht als endgueltige Vollstaendigkeit, weil spaetere modulare B-Strukturen (insbesondere die Antilinearitaet der modularen Konjugation J) weitere Bedingungen nachziehen koennen; der operative Produktivpfad bleibt `ExecComplex`, aber die Aussage „C wird erzwungen“ darf hier nur im ausgewiesenen Status gefuehrt werden	koppelt `PreNumericSectorCore`, `MatrixAlgebra`, `GeneralizedDtN` und `SectorExport`, ohne den Generator in eine ueberschossene Formalbehauptung umzuschreiben oder B-seitige Modullaritaet voreilig in A zu schliessen

Nr.	Modul / Bereich	Aktion	Muss lokal schließen	Muss wie verzahnen
40	`PillarA/Structural/CayleyDickson/*`, `PillarA/ScalarEmergence/QuaternionicSeed`	hinzufügen	Beweispflicht I als eigenen strukturellen Cayley-Dickson-Pfad ausserhalb von `PillarA/Handoff` schließen und zugleich die quaternionische Folgeoberflaeche darauf aufsetzen; Nichtkommutativitaet und bright/interface/dark-Lesart duerfen diagnostisch vorbereitet, aber nicht als bereits bewiesene Physik ausgegeben werden. Die Seed-Beschreibung muss explizit festhalten, dass auf der **H**-Stufe die Matrixalgebra ueber endlichen Carriern noch assoziativ bleibt, waehrend die tensoriell-kompositionelle Skalarseite an der Nichtkommutativitaet scheitert	trennt den strukturellen CD-Meta-Beweis sauber von der gerichteten Handoff-Schicht, liefert spaeteren Spin-/SU(2)-nahen Pfeilern eine saubere, vom operativen **C**-Produktivpfad getrennte Forschungs- bzw. opt-in Satzoberflaeche und verhindert, dass verschiedene Versagensmodi unter „Nichtkommutativitaet“ verwischt werden
41	`PillarA/ScalarEmergence/OctonionicSeed`, `PillarA/ScalarEmergence/HurwitzStopSeed`	hinzufügen	dritte Komplementierungsstufe, Alternativitaet/Nichtassoziativitaet und der durch Hurwitz begrenzte Stop der normierten Divisionsalgebren werden als externer Satz plus CNNA-interpretiertes Forschungsseed getrennt sichtbar gemacht; die Seed-Beschreibung muss explizit festhalten, dass auf der **O**-Stufe bereits die Matrixmultiplikation ueber endlichen Carriern nicht mehr assoziativ ist; zusaetzlich ist die Drei-Generationen-Lesart nur als benanntes Fernziel mit expliziten Abhaengigkeiten (mindestens **H**-Seed, **O**-Seed und spaetere DHR-/Matter-Seeds) zu fuehren; keine Identifikation „Universum = **O**“ im Generatorpfad	bereitet spaetere Matter-/Gauge-/Vereinheitlichungsdiskussionen vor, ohne OQS-, AQFT- oder Teilchenphysik frueh in A als erledigt auszugeben, trennt den Hurwitz-Randstein sauber von weitergehenden CNNA-Lesarten und laesst oktonionische Spezialoberflaechen nur explizit opt-in und randgebunden zu

13.1 Generatorengstellen und ihre Sollklassifikation

Objekt	Sollklassifikation	Zielmodul	Öffentliche Verzahnung	Migrationsschritt
`DirichletWeight`	legitimer A-seitiger Wurzelinput des universellen Pfads; auf Referenzpfaden zusätzlich kanonischer Default möglich	`Foundation/WeightPolicy`, konsumiert in `Finite/DirichletLaplacian`	`WeightPolicy.assign` bzw. ein `ofApproximant`-Builder; spätere Consumer sehen nur das freigegebene Policy-Gesetz, nicht freie Gewichtsfelder	freie Mittelstufen-Eingabe beseitigen; Gewichte an die Motorseite ziehen und optional kanonische Referenzpolicy für IDEAL/REAL-Referenzlauf ergänzen
`InteriorInverse`	kein Wurzelinput; Ziel ist algorithmische Ableitung als Solver-/Eliminationslogik des P12-Zahnrads	`DtN/DtN`	öffentliche Solve-, Randoperator- und Korrektheitssätze; ein eventuell verbleibender Inversionszeuge bleibt intern und lokal	nacktes Inversenfeld aus der öffentlichen Generatorverzahnung entfernen oder streng internalisieren; `noncomputable` nur am lokalen Solve-Rand akzeptieren

Objekt	Sollklassifikation	Zielmodul	Öffentliche Verzahnung	Migrationsschritt
`interfaceInverse`	kein Wurzelinput; Ziel ist algorithmische Ableitung als Restriktions-/Solver- bzw. Reduktionslogik der P22-Schur-Stufe	`Coupling/ MultiSchur`	öffentliche Schur-, Glueing-, Restriktions- und Korrektheitssätze; keine lose Inversenmatrix als Exportoberfläche	nacktes Inversenfeld aus der öffentlichen Generatorverzahnung entfernen oder streng internalisieren; Schur-Reduktion als echtes Zahnrad der P22-Fachlogik formulieren

13.2 Computability-Audit des Generators

Generatorzahnrad	Sollstatus	Akzeptanzkriterium	Bemerkung
Wurzel- und ToC-Seite bis `ToCStrong`	`computable`	alle öffentlichen `def`s kompilieren ohne `noncomputable`; Referenzfamilie und Variationsfamilie instanziieren den Pfad ohne `sorry`, ohne vermeidbare `noncomputable`-Kontamination und mit expliziten Instanzen für endliche Carrier dort, wo sie exponiert werden	hier darf keine schleichende `classical`-Abkürzung stehen bleiben
Finite Box-Seite bis `ApproximantStrong`	`computable`	alle exponierten Carrier, Teilmengen und Prädikate liefern `Fintype`-, `DecidableEq`- und `Decidable`-Instanzen; die öffentliche Oberfläche bleibt auswertbar; `grep` auf öffentliche `noncomputable def`s dieses Zahnrads muss leer sein	dies ist die Wurzel fast aller späteren Berechenbarkeitsfragen
MatrixNorms-Dualstrang `MatrixNorms`	operative Seite `computable`, analytischer Spiegelstrang nur nach Whitelist (d)	`frobeniusSq`, Nulltest, Positivitaet bei $\neq 0$ und die vom Generator konsumierte Shift-Oberflaeche laufen oeffentlich computable ueber `ExecComplex`; jede volle Frobenius-/Normspiegelung ueber $\mathbf{C}$ ist als parallele Nicht-Produktivinstanz gekennzeichnet und nur ueber die injektive Bruecke konsumierbar	`MatrixNorms` ist der erste Normalfall der Whitelist-Klasse (d) und der einzige legale Zulieferer der computablen Shift-/Vergleichsseite fuer `DtNStabilized`

Generatorzahnrad	Sollstatus	Akzeptanzkriterium	Bemerkung
Algebraische A-Wurzeln `HermitianStructure/FiniteHilbert/MatrixAlgebra`	`computable`	Adjungierte, innere Produkte, $*$-Algebra, Kommutator und elementare algebraische Seeds stehen öffentlich ohne `noncomputable`; die öffentliche Produktivinstanz läuft über einen explizit ausführbaren Koeffiziententyp (Vorbereitungsblock `ExecComplex`), während die spätere **C**-Erweiterung nur über ein eigenes Brückenmodul als paralleler Spiegelstrang erfolgt; alle späteren finite Seeds konsumieren operativ den `ExecComplex`-Strang statt eigener Hilfslogik oder stillschweigender **C**-Spezialpfade	verhindert versteckte algebraische Basisarbeit in `SpectralDecomposition`, `StateSpace` oder B/C und schützt den Generator vor analytischer Frühkontamination
Operatorseite `DirichletLaplacian/DtNOnStable`, `DtNStabilized`	öffentlich `DirichletWeight` läuft über `WeightPolicy`; eventuell verbleibende Solve-/Inversionszeugen bleiben intern und theorematisch abgeschirmt; jede öffentliche `def` kompiliert ohne `noncomputable`; `DtNStabilized` konsumiert Shift- und Vergleichsgrößen ausschließlich aus der computablen `ExecComplex`-Seite von `MatrixNorms`	akzeptiert werden nur klar lokalisierte Restgrenzen, kein diffuser Globalstatus des Generators und kein versehentlicher Rueckgriff auf den **C**-Spiegelstrang
Spektral-, Zustands-, thermische und dynamische Seeds	öffentlich `computable`, intern höchstens lokal `noncomputable`	`SpectralDecomposition`, `StateSpace`, `MatrixExponential`, `GibbsStateSeed`, `UnitaryEvolution` und `DynamicsAdapter` exponieren nur kanonische, auswertbare Oberflächen; etwaige interne Nichtberechenbarkeit muss aus der Whitelist begründet werden. Fuer `MatrixExponential` ist die Designentscheidung explizit zu dokumentieren: im `ExecComplex`-Pfad ist zunächst nur eine endliche, computable Partialsummen- bzw. Approximationsoberfläche zulässig; der Zusammenhang zum analytischen Exponential über **C** wird erst über die Brücke theoremiert und nicht stillschweigend vorausgesetzt	diese Seeds dürfen nicht erst in B/C nachgebaut werden
Coupling- und Handoff-Seite bis `PillarAStep1Closed`	öffentlich `computable`, intern höchstens lokal `noncomputable`	Schur-/Glueing-/Handoff-Oberflächen sowie IDEAL/REAL-Provenienz bleiben von internen Solverdetails getrennt; spätere Pfeiler konsumieren eine kanonische, möglichst computable Exportfläche; `ABHandoffStrong` bleibt reiner Auswärtsadapter	falls lokale Nichtberechenbarkeit unvermeidbar bleibt, ist sie im Modul und im Plan explizit zu benennen

Whitelist für verbleibende `noncomputable`-Reste im aktiven Pfad. Jede `noncomputable def`

in einem Strengthening-Modul muss explizit genau einer der folgenden Klassen zugeordnet werden:

- **(a) lokale Matrixinversion / Solve-Logik:** ein interner Inversions- oder Solverzeuge, der öffentlich nur über Korrektheits- und Eliminationssätze exponiert wird;
- **(b) IdealAddressEquiv- oder vergleichbarer Transport über genuinely unendlichem Träger:** ein nicht-algorithmischer Äquivalenz- bzw. Transportrest der IDEAL-Seite;
- **(c) genuinely unendliche Filter-/Limesselektion:** ein unvermeidbarer `classical`-Rest auf der IDEAL- oder Directed-Limit-Seite, der nicht in öffentliche finite Seeds oder Handoffs einsickern darf;
- **(d) C-seitige Parallelinstanz mit Bridge-Transfer:** eine `noncomputable def`, die ausschliesslich als mathlib-**C**-Spiegelung einer bereits computable geschlossenen `ExecComplex`-Definition existiert, nur ueber die bewiesene injektive Bruecke konsumiert wird und **nicht** als eigenstaendige oeffentliche Produktionsoberflaeche des Generators dienen darf.

Alles andere gilt im Strengthening als Planverstoß und ist vor Phasenschluss zu beseitigen. Die Whitelist-Klasse (d) ist kein Freibrief fuer einen zweiten Generatorpfad, sondern dient ausschliesslich dazu, den analytischen Spiegelstrang auditierbar vom operativen Produktivstrang zu trennen.

14 Strengthening-Phasen S0–S20 mit Unterphasen S6a/S6b/S6c, S7a/S7b, S8a/S8b/S8c/S8d/S9a/S9b/S9c/S9d/S9e/S9f/S9g/S9h, S10a/S10b/S10c und S11a/S11b

Hinweis zum gruenden Arithmetikstand A1–A6. Der nach gruener A6-Schliessung erreichte A-interne Arithmetikpfad unter `PillarA/Arithmetic/*` wird im vorliegenden Plan als eigener Folgeblock gemaess Abschnitt 15 gefuehrt. Er ist weder ein zweiter Root-Generator noch ein Ersatz fuer den S-Hauptpfad; insbesondere zieht er S10a nicht vor, ersetzt nicht Beweispflicht II und konsumiert keine Handoff-Ausgangsobjekte als internen Abkuerzungsweg.

Phase	Betroffene Module	Muss am Phasenende wahr sein
S0	Plan-Neuschnitt, betroffene `BuildAll`- und `Notation`-Dateien, Repo-Audit	Strengthening als Schritt 2 ist terminologisch und architektonisch sauber vom abgeschlossenen Refactor getrennt; zugleich ist durch Code-Audit explizit festgehalten, welche Builder bereits stehen, wo die realen Generatorengstellen sitzen, welche algebraischen Seeds spätere Pfeiler wirklich voraussetzen und dass Export/Handoff als reine Kupplung späterer Pfeiler zu lesen sind
S1	`Foundation/SubstrateClass`, `Foundation/MatrixNorms`, `Foundation/Interfaces`, `Foundation/Determinism`, `Foundation/HermitianStructure`, `Foundation/FiniteHilbert`, `Foundation/MatrixAlgebra`, `Foundation/ConcreteSubstrate`; abgeschlossener Vorbereitungsblock V0–V5 (algebraische Seite gruen; konkrete Referenzfamilie in S1 weiterhin separat offen)	die typklassenparametrische Universalität bleibt Wurzel des Produktivpfads; zugleich steht eine computable konkrete Referenzfamilie ohne `sorry` und ohne vermeidbare `noncomputable`-Kontamination, `MatrixNorms` ist als erster expliziter Dualstrang-Consumer legal umgestellt, die algebraische Matrix-/Hilbert-/$*$-Grundlage ist über eine öffentlich computable Wurzel mit vorbereiteter strukturtreuer Brücke ExecComplex $\to$ **C** ausgerichtet, und `Foundation/Interfaces` exponiert bereits die generischen Netz- und Vertragsobjekte, die spätere Pfeiler nur noch instantiieren dürfen
S2	`Foundation/LevelVariableSubstrate`, `Foundation/WeightPolicy`, `Foundation/RegularizationPolicy`, `Foundation/SubstrateAnalysis`	eine zweite legale Substratfamilie, eine explizite `WeightPolicy`-Achse und eine numerisch-kanonische `RegularizationPolicy`-Achse stehen; zugleich bleibt β als sichtbare thermische Provenienz- und Variationsachse erhalten und ist auf dem produktiven Generatorpfad zunaechst als rationaler Parameter modelliert, waehrend die spaetere **R**-/**C**-Interpretation nur ueber die Bruecke erfolgt; ausserdem ist auditierbar gemacht, welche Aussagen universell über `SubstrateClass` laufen und welche stärkere Voraussetzungen benötigen
S3	`ToC/Contract`, `ToC/Addressing`, `ToC/IdealAddressEquiv`, `ToC/ConcreteIdeal`	der konkrete Referenzfall des IDEAL-ToC ist als Familie geschlossen und legal an Contract/Addressing/Equiv rückgebunden; die zugehörige Referenz-/Aliasoberfläche ist definiert und importseitig freigegeben, ihre operative Konsumption im gemeinsamen realen Generatorpfad wird jedoch erst ab S4 verpflichtend mitgezogen; deterministische Zusätze bleiben flankierend und werden nicht heimlich zum Generatorvertrag erklärt

Phase	Betroffene Module	Muss am Phasenende wahr sein
S4	`ToC/Concrete`, `Finite/CutSpec`, `Finite/RegionCore`, `Finite/BoundaryPorts`, `Finite/Approximant`, optional `Finite/Selection`	Referenzlauf und Variationslauf laufen nun ab dem ersten realen `ToCStrong`-/Approximantenkern in denselben endlichen Box-Pfad ein; die exponierten Carrier und Praedikate liefern die noetigen `Fintype`/`Decidable`-Instanzen; die in S3 freigegebene Referenznotation wird im realen Downstream sichtbar konsumiert und bleibt nicht bloss global gespiegelt; der harte Typcheck fuer beide Familien belegt, dass kein Smoke-Test-Sonderpfad entsteht. Kleine nicht blockierende Nachzuege bleiben als Audit notiert: ein unnoetiger `Determinism`-Import in `ToC/Concrete` ist noch zu bereinigen bzw. spaeter sichtbar zu konsumieren, und die Variationsseite ist notationsmaessig bis spaetestens S12 noch symmetrischer zu spiegeln
S5	`Finite/DirichletLaplacian`, `DtN/DtN`	die beiden ersten Generatorengstellen sind geschlossen: `DirichletWeight` ist derived-only an `ApproximantStrong`/`WeightPolicy` gebunden, die Referenz-/Variationsfamilie bleibt auch auf der ersten Operatorstufe im selben Produktivpfad, und die DtN-Seite exponiert keine freie oeffentliche Inverseneingabe mehr, sondern eine proof-carrying Solver-/Eliminationsoberflaeche mit Randoperator, Innen-Solve und Korrektheitssaetzen
S6a	`DtN/DtNStabilized`, `Closure/ParameterClosure`, `Closure/RegularizationClosure`	die abgeschlossene Shift-, Stabilisations- und Closure-Seite ist soweit legal möglich internalisiert; primitive Restparameter werden nicht mehr unkommentiert durch den Produktionspfad getragen, und `DtNStabilized` konsumiert die noetigen Shift-/Vergleichsgroessen ausschließlich aus der computablen Produktivseite von `MatrixNorms` statt aus einer analytischen **C**-Spiegelinstanz
S6b **(abgeschlossen)**	`PillarA/Structural/CayleyDickson/*`, gegebenenfalls duenner Abschluss in `Handoff/ProofObligationI/*`	Beweispflicht I wird als eigener A-interner struktureller Cayley-Dickson-Pfad formal geschlossen: der volle CD-Beweiskoerper lebt innerhalb von Pillar A, aber ausserhalb von `PillarA/Handoff`, und ist nicht als normaler Inter-Pillar-Kanal missverstaendlich. Im gruenden Abschlussstand sind damit die zugehoerigen Slot-1/2/3-Pflichten von Duty I in ihrer aktuellen formalen Fassung bewiesen; insbesondere sind carrier, kanonische Multiplikation, die normbezogene Pflichtoberflaeche samt struktureller Kreuzterm- bzw. Kompositionsanalyse und die daraus benoetigte Divisionslogik auf dem S6b-Pfad proof-carrying geschlossen. Falls eine Handoff-seitige Sicht gebraucht wird, erscheint dort nur eine duerre, gerichtete Abschluss- oder Ledger-Oberflaeche; der interne Beweis selbst darf weder als Exportersatz noch als Rueckkanal dienen. Eine spaetere semantische Nachschaerfung von Slot 2 im Scalar-Emergence-Block bleibt davon getrennte Folgearbeit und ist kein Blocker fuer das nachfolgende Handoff-Refactoring in S6c
S6c **(abgeschlossen)**	`PillarA/Handoff/Core/*`, `PillarA/Handoff/Outputs/*`, `PillarA/Handoff/Inputs/*`, `PillarA/Handoff/ProofObligationI/II/III/*`	Die Handoff-Seite ist im gruenden Abschlussstand nach der abgeschlossenen S6a-Closure und dem CD-Pfad aus S6b neu geordnet: keine flache Sammelablage mehr, sondern gerichtete Schnittstellensemantik. `Outputs` sind reine Exportobjekte, `Inputs` die einzigen zulaessigen Backreaction-Kanaele, und `ProofObligationII` ist strukturell so vorbereitet, dass ab S10a inhaltlich begonnen werden kann; `ProofObligationIII` bleibt als spaetere, theorie-erzwungene Rueckwirkung von B/C nach A vorbereitet. Damit ist S6c planlogisch geschlossen und S7a als naechster Kopplungsschritt freigegeben
S7a **(abgeschlossen)**	`Coupling/GeneralizedDtN`, `Coupling/MultiSchur`	die Mehrsektor-Kopplung ist im gruenden Abschlussstand familiesicher geschaerft: `GeneralizedDtNStrong` traegt raw/stabilized explizit getrennt als theorematisch gebundene Operatoroberflaechen derselben legalen P21-Provenienz, `MultiSchurStrong` konsumiert operativ ausschliesslich den stabilisierten Restriktionspfad, und `interfaceInverse` erscheint nicht mehr als nackter Wurzelinput, sondern nur noch als innerhalb von P22 gebundene Interface-Eliminations- bzw. Reduktionslogik. Damit ist S7a planlogisch geschlossen und S7b als naechster spaeter Traeger-Schritt freigegeben

Phase	Betroffene Module	Muss am Phasenende wahr sein
S7b **(abgeschlossen)**	`Network/InfiniteCarrier`, begleitend `Network/SectorChannels`	die späte Träger-/Kanalstufe ist im gruenden Abschlussstand familiesicher an den bereits geschlossenen Netz- und Kopplungspfad gebunden: `InfiniteCarrierStrong` exportiert stabile Ausschnitte und gerichtete Uebergaenge weiterhin proof-carrying fuer Vergleiche und spaetere Limesdiagnostik, ohne eine neue ontische Unendlichkeitsquelle einzufuehren, und `SectorChannelsStrong` bleibt flankierender Supportconsumer derselben legalen Provenienz, ohne einen eigenen semantischen Seitenpfad zu eroeffnen. Damit ist S7b planlogisch geschlossen und S8a als erste Unterphase der naechsten endlichen Spektralwurzelphase freigegeben
S8a **(abgeschlossen)**	`Finite/SpectralDecomposition`, `Finite/SpectralDecompositionC`, `Finite/SpectralBridge`, begleitend `Finite/BuildAll`, `Finite/Notation`, `PillarA/Notation`	der S8-Einstieg ist im gruenden Abschlussstand als expliziter Dualstrang-Architekturflip geschlossen: `Finite/SpectralDecomposition` ist die einzige oeffentliche operative `ExecComplex`-Oberflaeche der Spektralwurzel und exponiert zunaechst nur auswertbare Matrix-, Hermitizitaets-, Frobenius- und Shift-nahe Seeds ueber demselben A-seitig erzeugten Dirichlet-Operator; `Finite/SpectralDecompositionC` traegt davon getrennt die analytische **C**-Spiegelinstanz nur als zulaessige Whitelist-(d)-Parallelschicht; `Finite/SpectralBridge` bindet beide Seiten ueber eine explizite injektive Bruecke, sodass der Spiegelstrang weder zum zweiten Generatorpfad noch zur stillschweigenden oeffentlichen Produktionsoberflaeche werden kann. Damit ist die S8-Architektur bereinigt, der gruene Build haelt die neue Konsumrichtung bereits aus, und S8b kann nun auf dieser operativen Exec-Wurzel aufsetzen
S8b **(abgeschlossen)**	`Finite/SpectralDecomposition`, begleitend `Finite/Notation`, `PillarA/Notation`	die oeffentliche operative Spektraloberflaeche ist im gruenden Abschlussstand ueber dem `ExecComplex`-Strang zu einer echten proof-carrying Minimalwurzel verschaerft: kanonische spektrale Kandidaten-, Schatten- und Zertifikatsdaten des endlichen Dirichlet-Operators sind auswertbar vorhanden, oeffentliche `noncomputable`-Definitionen bleiben verboten, und spaetere Seeds koennen Spur-, Determinanten-, Charakteristikum- und erste Projektor-/Kommutationszertifikate konsumieren, ohne auf den analytischen Spiegelstrang auszuweichen. Der Abschluss enthaelt dabei bewusst nur eine enge vorbereitende Vorziehung eines kleinen Teils des urspruenglich in S8e sichtbaren `ExecSpectral`-Folgeblocks: nicht der allgemeine computable Hermitesch-Pfad ist bereits geschlossen, wohl aber genau die minimale operative Vorstufe, die fuer eine legal geschlossene proof-carrying Exec-Wurzel noetig war
S8c **(abgeschlossen)**	`Finite/SpectralDecompositionC`	der analytische **C**-Spiegelstrang ist im gruenden Abschlussstand lokal und auditierbar vervollstaendigt: die mathlib-seitige hermitesche Spektralzerlegung des gespiegelt eingebetteten Dirichlet-Operators steht als explizit markierte Whitelist-(d)-Parallelinstanz mit Eigenwerten, Eigenvektorbasis, unitaerer Diagonalisierung und Spektralsatz proof-carrying bereit, bleibt aber streng vom operativen Produktivpfad getrennt und darf nur ueber die Bruecke konsumiert werden. Der Abschluss wurde dabei ausdruecklich ueber explizite strukturelle Beweise und eine enge lokale `noncomputable`-Insel im analytischen Spiegelpfad realisiert, ohne die oeffentliche `ExecComplex`-Produktivoberflaeche zu oeffnen. Damit ist S8c planlogisch geschlossen und S8d als naechste sichtbare Brueckenphase freigegeben
S8d **(abgeschlossen)**	`Finite/SpectralBridge`	die Brueckenstufe ist im gruenden Abschlussstand proof-carrying geschlossen: operative `ExecComplex`-Oberflaeche und analytischer **C**-Spiegel sind theorematisch als derselbe A-seitig erzeugte Operator gebunden; Matrixidentifikation, Hermitizitaetstransfer sowie die kontrollierte Uebertragung von Spur-, Determinanten-/Charakteristikum-, Diagonal- und ersten projektoriellen Schatten laufen explizit ueber die injektive Bruecke. Der analytische Spiegel bleibt auditierbarer Parallelstrang und darf von spaeteren Consumern nicht als operative Produktionsoberflaeche missbraucht werden. Damit ist S8d planlogisch geschlossen und S8e auf den verbleibenden allgemeinen computablen `ExecSpectral`-Block verengt

Phase	Betroffene Module	Muss am Phasenende wahr sein
	`Finite/ExecSpectral/PolynomialCore`, `Finite/ExecSpectral/CharpolyCore`, `Finite/ExecSpectral/RootIsolation`, `Finite/ExecSpectral/EigenvectorKernel`, `Finite/ExecSpectral/ProjectorCore`,	
S8e **(abgeschlossen)**	`Finite/ExecSpectral/Certificates`	der nach S8b und S8d verbleibende allgemeine computable `ExecSpectral`-Folgeblock ist im gruenden Abschlussstand als sichtbare Restschicht proof-carrying geschlossen: `PolynomialCore`, `CharpolyCore`, `RootIsolation`, `EigenvectorKernel`, `ProjectorCore` und `Certificates` tragen nun die verbleibende computable Polynomarithmetik-, Charakteristikums-, Wurzelisolations-, Kerneigenvektor-, projektorielle und Zertifikatsoberflaeche des operativen `ExecComplex`-Strangs, ohne die bereits in S8b absorbierte Minimalwurzel oder die bereits in S8d geschlossene Bruecken- und Spiegeltransferklasse zu duplizieren. Damit ist S8e planlogisch abgeschlossen; S8f ist nun der aktive naechste Schritt und muss diese sichtbare Restschicht ausschliesslich als Consumer zu einer moeglichst allgemeinen algebraischen Spektralwurzel der oeffentlichen operativen Oberflaeche zusammenschliessen
S8f **(abgeschlossen)**	`Finite/SpectralDecomposition`, begleitend `Finite/ExecSpectral/*`, `Finite/SpectralBridge`	die eigentliche Spektralwurzelphase ist im gruenden Abschlussstand planlogisch geschlossen: die oeffentliche operative `SpectralDecomposition`-Oberflaeche konsumiert den zuvor sichtbaren `ExecSpectral`-Folgeblock nun selbst, exportiert die verfuegbare algebraische Spektraloberflaeche des `ExecComplex`-Strangs proof-carrying und laesst keinen zweiten analytischen Generatorpfad zu. Verbleibende Reichweiten- oder Regimefragen sind damit nicht mehr Teil von S8, sondern gehoeren als strikt getrennte Folgearbeit in S9 sowie die spaeteren Diagnose-, Regime- und Statusbloecke; insbesondere werden weder Kesten–McKay-Recovery, noch Gap-als-Masse-Lesarten, noch RG- oder Normpflichten als S8g+ in die geschlossene Spektralwurzelphase zurueckgezogen. Damit ist S8 insgesamt abgeschlossen, und S9 bleibt der naechste aktive Phaseinstieg
	`Finite/StateSpace`, `Finite/MatrixExponential`, `Finite/GibbsStateSeed`, `Finite/UnitaryEvolution`, `Finite/DynamicsAdapter`,	
S9a	`Finite/ChannelSeed`	das gruende S9-Grundgeruest ist zunaechst nur als Audit-Basis sauber klassifiziert: fuer jede oeffentliche Definition ist explizit entschieden, ob sie auf den operativen `ExecComplex`-Pfad gehoert, als auditierbare **C**-Mirror-/Bridge-Flaeche sichtbar bleiben darf oder nur internes Hilfsgeraet ist. Der aktuelle Codebefund – insbesondere die noch vorhandenen 28 oeffentlichen `noncomputable def` ueber die sechs S9-Module – gilt dabei explizit *nicht* als zulaessiger Endzustand; verbleibende Whitelist-Reste muessen datei- und definitionsscharf benannt sein, und es entsteht weder ein neuer analytischer Generator noch ein semantischer Seitenpfad neben der bereits geschlossenen S8-Spektralwurzel
S9b	`Finite/StateSpace`	`StateSpaceStrong` exponiert oeffentlich zuerst eine computable operative Zustandsoberflaeche ueber `ExecComplex`: `execDensityOperator`, `execTrace`, `execTraceNormalize` sowie basisnahe Exec-Zustaende und Exec-Projektoren sind proof-carrying geschlossen. Die bisherigen komplexen Begriffe `complexDensityOperator`, `trace`, `traceNormalize`, `IsPositive/positiveCone` und der projektorielle Zugriff ueber den komplexen Koordinatenprojektor bleiben hoechstens als explizit benannte Mirror-/Bridge-Sektion sichtbar; ein oeffentlicher Fallback der Form `if Z = 0 then ... else ...` ist als Endzustand unzulaessig und muss entweder durch einen Nichtnull-Zeugen oder durch einen bewiesenen Nichtnull-Satz fuer die relevanten Gibbs-/Projektorfalle ersetzt werden
S9c	`Finite/MatrixExponential`	`MatrixExponentialStrong` ist im operativen Produktivpfad von der aktuellen oeffentlichen Platzhalterform `NormedSpace.exp` geloest: oeffentlich exportiert werden nur eine endliche computable Partialsummen-/Approximationsoberflaeche `expApprox` samt `scaledGeneratorApprox`, `evolutionApproxAt` und `thermalApproxAt` ueber dem `ExecComplex`-Strang. Die analytische **C**-Exponentialfunktion bleibt hoechstens Mirror-/Bridge-Flaeche; zugleich ist die thermische Provenienz des operativen Pfads sichtbar an `WeightPolicy/ThermalAxis` mit rationalem β gebunden statt ueber eine nackte reelle Generatoroberflaeche zu laufen

Phase	Betroffene Module	Muss am Phasenende wahr sein
S9d	`Finite/GibbsStateSeed`	der oeffentliche Gibbs-Seed ist vom analytischen Spiegel geloest und laeuft derived-only ueber die S9-Leiter selbst: `gibbsApproxUnnormalized` und `gibbsApproxState` konsumieren die operative Approximation aus `MatrixExponential`, die thermische Standardoberflaeche ist an `WeightPolicy.thermal` rueckgebunden, und die Nichtnullheit der Partitionsfunktion ist theorematisch gesichert oder explizit als benoetigter Zeuge exponiert. Direkte oeffentliche Abhaengigkeiten von `mirror.eigenvalues` oder `mirror.eigenvectorUnitaryMatrix` bleiben nur als klar getrennte Analyse-/Bridge-Flaeche zulaessig
S9e	`Finite/UnitaryEvolution`	die dynamische Oberflaeche ist fachlich und operativ geschaerft: `schrodingerConjugation` und `heisenbergConjugation` sind nicht mehr bloss dieselbe Formel unter anderem Namen, sondern konsumieren `propagator` bzw. `inversePropagator` richtungsrein; Nullzeit- und Kompositionssaetze sind explizit nachgezogen. Falls ein nackter reeller Zeitparameter den computablen Produktivpfad oeffnet, ist bis Phasenschluss eine explizite computable Approx-Achse oder ein geeigneter Zeitparametertyp einzuziehen
S9f	`Finite/DynamicsAdapter`	`DynamicsAdapterStrong` ist nicht mehr nur duenne Verpackung um `UnitaryEvolution`, sondern eine proof-carrying `StarAlgDynamics`-Vorstufe: Zustand, Dichte und Observable sind strukturell und theorematisch aufeinander bezogen, Nullzeit, Komposition und gegebenenfalls Adjungiertenvertraeglichkeit sind nachgezogen, und `starAlgDynamicsSeed` bleibt nicht auf einen blossen Dreifach-Record aus Maps reduziert
S9g	`Finite/ChannelSeed`	die algebraische Kanal-Vorstufe ist ueber die aktuelle duenne Package-Schale hinaus zu einer echten Kraus-/Choi-/CPTP-Anschlussoberflaeche ausgebaut: `AlgebraicChannelPackage` traegt `choiMatrix`, `isTracePreserving`, `isPositive`, `isCompletelyPositive` und `isCPTP`; fuer den unitären Spezialfall steht mindestens ein expliziter Single-Kraus-Seed, und Identitaet, beidseitige Nullzeit sowie Komposition sind theorematisch gebunden. Damit muessen spaetere Pfeiler keine rohe Superoperator-Basis neu aufbauen
	alle sechs S9-Module, `Finite/BuildAll`, `Finite/Notation`, `PillarA/Notation`,	
S9h	Audit-/Plantext	jetzt erst darf S9 als geschlossen markiert werden: ein Audit auf oeffentliche `noncomputable def`, neue `classical`-Reste, pauschale `@[simp]`-Flaechen, Importketten und real konsumierte Notation ist build- und grep-pruefbar gruen. Der harte Abschlussmassstab lautet: oeffentliche operative S9-Oberflaeche computable, Mirror-Inseln nur lokal und explizit, `MatrixExponential` ueber Partialsummen statt nacktem `NormedSpace.exp`, Gibbs nicht mehr ueber die oeffentliche Mirror-Eigenzerlegung, Heisenberg/Schroedinger sauber getrennt und `ChannelSeed` mit echter Kraus-/Choi-/CPTP-Basis geschlossen
S10a	`Network/SectorSysEnv`, `Network/RelativeEntropyFlow`, `Geometry/LCPMeasure`, `Geometry/Foliation`, `Geometry/SpacetimePaths`, `Network/DirectedLimit`; `Network/RegionNet`, `Network/SectorChannels` nur unverändert als legale Vorgänger	die nicht-kernblockierende Netzwerk-, Geometrie- und Limesvorstufe steht; IDEAL- und REAL-Provenienz bleiben getrennt sichtbar, und `DirectedLimit` exportiert eine PreNet-kompatible Übergangsfamilie, ohne bereits AQFT-Limessemantik zu behaupten. Zugleich ist die strukturelle Vorbedingung geschaffen, um Beweispflicht II ab hier kontrolliert zu beginnen
S10b	`PillarA/VariationAnalysis`, `PillarA/Diagnostics/SpectralDimension`, `PillarA/Diagnostics/SymmetryCutoffLimits`, `PillarA/Diagnostics/ScaleBreaking`, `PillarA/Diagnostics/ApproxSymmetry`, `PillarA/Update/MismatchSeed`, `PillarA/Update/TailEliminationSeed`, `PillarA/Update/UpdateStep`	die nicht-kernblockierende Diagnose-, Variations-, Scale-Breaking-, Symmetrie- und Update-Seed-Schicht steht; Spektraldimensionsfluss sowie Symmetrie-, Cutoff- und Limesdiagnostik sind jeweils explizit in IDEAL- und REAL-Seite getrennt A-seitig vorbereitet, `VariationAnalysis` klassifiziert Theoreme nach Universalitäts-, Substrat-, Policy-, ϵ-/β- und Cutoff-Abhängigkeit, und die Exportdiagnostik unterscheidet bereits zwischen Einzelsektoroberflaechen, expliziter Kopplungsprovenienz und solchen effektiven Randdaten, die als Projektion reell erscheinen koennen, ohne die zugrunde liegende Kopplungsrichtung selbst schon voll auszumodellieren

Phase	Betroffene Module	Muss am Phasenende wahr sein
	`PillarA/Diagnostics/DualObserverSeed`, `PillarA/Diagnostics/InterfaceEntropyBound`, `PillarA/Diagnostics/SpectralGapSeed`, `PillarA/Diagnostics/SchurBlockSymmetrySeed`, `PillarA/Diagnostics/HorizonDefinition`,	
S10c	`PillarA/Diagnostics/CayleyDicksonSpectralShift`	die algebraische und spektrale Regimevorstufe fuer spaetere Horizont-, Projektions- und Casimir-Kandidaten ist A-seitig derived-only vorbereitet: `DualObserverSeed` formalisiert die duale Equivarianz der Seed-Kette, `InterfaceEntropyBound` liefert ein entropisches Boundary-Carrier-Mass, `SpectralGapSeed` den kleinsten positiven DtN-Eigenwert nur als expliziten Masse-/Kopplungskandidaten der CNNA-internen Seed-Sprache, `SchurBlockSymmetrySeed` die duale Symmetrie der Schur-Bloecke, `HorizonDefinition` das Kriterium $\lambda^{+}_{\min} \to 0$, und `CayleyDicksonSpectralShift` die spektrale Verschiebung entlang der CD-Fortsetzung. Diese Stufe importiert keine B/C/D/E-Fachlogik, schliesst weder Casimir-, Feynman- noch Hurwitz-Physik und auch keinen externen Universalitaetssatz zu Kesten–McKay oder RG-Fixpunkten; die gestern zusaetzlich genannte Bezeichnung `ProjectionRegimeSeed` bleibt nur als moegliche spaetere duenne Wrapper- oder Aliasoberflaeche ueber `SpectralGapSeed` plus `HorizonDefinition` offen
S11a	`Handoff/BInterfaceSeeds`, `Handoff/SectorExport`, `Handoff/Step1StrongCore`	die zielneutrale Kupplungs- und Verdichtungsstufe exportiert Generator-, algebraische, spektrale, zustandsräumliche, thermische, dynamische, kanalische und IDEAL/REAL-Diagnoseprovenienz einmalig nach außen; dabei muessen Einzelsektor-, Kopplungs-, Regularisierungs-, raw/stabilized-, IDEAL/REAL- und Substratvariations-Provenienz getrennt sichtbar bleiben, damit spaetere Pfeiler keine A-seitigen Vergleichsachsen neu konstruieren muessen. Zugleich stehen konstruktive BInterface-Seeds aus den reifen A-Daten bereit, und `SectorExport` wird nicht doppelt in späteren Phasen angefasst
S11b	`Handoff/Step1MathData`, `Handoff/ABHandoffStrong`, `Handoff/PillarAStep1Closed`, `PillarA/Generator`	der produktive Root-to-Handoff-Generator steht; Referenzfall und beliebige legale Kombination aus Substratfamilie, `WeightPolicy`, `RegularizationPolicy`, sichtbarer β-Achse und Cutoff erzeugen über denselben Pfad ein echtes `PillarAStep1Closed`. Zugleich ist das Computability-Budget nach Whitelist auditiert, die öffentliche Generatoroberfläche bleibt soweit wie möglich computable, `ABHandoffStrong` exponiert den A-seitigen `StarAlgebraContract` nur als ausgehenden Instantiierungssatz, und jede legale Generatorinstanz liefert damit bereits den spaeteren Vergleichsursprung fuer Spezialregime-, Reduktions-, Grenzfall- und Theorievergleichsfragen
S12	alle betroffenen `Notation`-Dateien	der gesamte Strengthening-Pfad ist lesbar notiert: Referenzfamilien, algebraische Seeds, Gibbs-/Dynamik-/Update-Seeds, BInterface-Seeds, Generator, Directed-Limit und Diagnosemodule sind scoped und parameter-sichtbar, ohne Transporte oder Provenienz zu verdecken
S13	alle betroffenen `BuildAll`-Dateien, Root-Build	der Strengthening-Pfad ist vollständig in den aktiven Build eingehängt; keine zweite Schatten-Topologie und kein Rückgriff auf Archivpfade verbleiben
S14	Root-Build, Handoff-Audit zwischen den Pillars, finaler Generator- und Exportaudit	der Generator-Kernblock des Strengthening-Pfads ist produktiv rauchtesttauglich und zugleich streng pfeilerisoliert: spätere Pfeiler konsumieren nur die bereitgestellten Export- und Handoff-Flächen, ohne rückwirkend frühere Pfeiler zu reparieren oder algebraische A-Vorstufen neu zu implementieren. Spätestens hier muss auditiert sein, dass alles, was fuer spaetere Spezialregimevergleiche aus A heraus vorbereitbar ist, exportseitig sichtbar gemacht wurde
S15	`PillarA/ScalarEmergence/PreNumericSectorCore`, `PillarA/ScalarEmergence/StatusLedger`	der interne Scalar-Emergence-Folgeblock ist formal geoeffnet: die prae-numerische Sektoroberflaeche ist als eigener theorematischer Consumer sichtbar, und der Plan fuehrt explizit Buch darueber, was bereits maschinengeprueft, was externer Satz, was strukturelle Beobachtung, was komparative Forschungslesart, was offene Frage und was Fernziel ist. Dazu gehoeren pro Stufe kurze Notizen der konkreten Versagensart, getrennte externe Anker (mindestens Hurwitz, Solèr 1995, Barnum/Müller/Ududec 2014, Renou et al. 2021 sowie fuer spaetere Spektral-Regimefragen explizit Kesten 1959 und McKay 1981) sowie benannte Fernziele mitsamt Abhaengigkeiten und Literaturankern; insbesondere werden REALOQS-Ruecklese, LQG-/Quaternionenvergleich, Kesten–McKay-Recovery, RG-Lesarten und exzeptionelle Horizonte nur statusmarkiert und nicht mit Generatorclosure vermischt

Phase	Betroffene Module	Muss am Phasenende wahr sein
S16	`PillarA/ScalarEmergence/OneSectorRealSeed`, flankierend `Foundation/WeightPolicy`, `Finite/StateSpace`, `PillarA/ScalarEmergence/ComplexCouplingSeed`	ein ontische Einzelsektor-Schicht ist planlesbar geschlossen: Bei leerem dunklem Carrier bzw. `noOuterEnvironment` werden reelle bzw. reell-positive Einzelsektor-Daten als eigenes Paket gebuendelt; Positivitaet, Gewichte, Normierung, Symmetrie, reelles Spektrum und die triviale Degeneration des dunklen Eliminationsblocks stehen als $\mathbf{R}_{\geq 0}$-artige Vorstufe bereit, ohne bereits Komplementkopplung oder Phaseninformation zu unterstellen. Zugleich wird explizit auditiert, dass das Interface in dieser Lage nicht automatisch verschwindet, sondern mit der bright-Grenze zusammen gelesen werden muss. REALOQS darf rueckblickend als Einzelsektor- bzw. effektiv reelle Vergleichsoberflaeche gelesen werden; diese Ruecklesart gilt jedoch zunaechst nur als statusmarkierte Interpretation von Legacy-Code und nicht als abgeschlossener A-Satz
S17	(Phasenlesart: *ComplexEmergenceBridge*), flankierend `Foundation/ExecComplexBridge`, `Coupling/GeneralizedDtN`, `Handoff/SectorExport`, `PillarA/ScalarEmergence/QuaternionicSeed`; A-interner CD-Pfad aus S6b als legaler upstream-Vorgaenger	die operative **C**-Kopplungsschicht ist nun nicht nur technisch, sondern semantisch ausgewiesen: die vier Bedingungen Interferenz, Reversibilitaet, Positivitaet und Kompositionalitaet sind als explizite Constraint-/Hypothesenoberflaeche formalisiert; zusaetzlich wird die Lesart vorbereitet, dass komplexe Phase, gerichteter Interface-Fluss und Involution strukturell zusammengehoeren koennten. Zugleich ist festgehalten, dass diese Liste vorlaeufig nur die A-seitige Mindestliste bildet und spaetere modulare B-Daten – insbesondere die Antilinearitaet der modularen Konjugation J – weitere Bedingungen nachziehen koennen; klar auditiert bleibt, dass ein Zwangssatz auf **C** hier noch nicht als maschinengepruefter Abschluss vorliegt
S18	`PillarA/ScalarEmergence/OctonionicSeed`, `PillarA/ScalarEmergence/HurwitzStopSeed`; A-interner CD-Pfad aus S6b als legaler upstream-Vorgaenger	die zweite Komplementierungsstufe ist als eigener A-interner Folgepfad vorbereitet: Nichtkommutativitaet, bright/interface/dark-Lesart und quaternionische Folgeinterpretationen werden explizit gefuehrt, ohne den operativen **C**-Generatorpfad oder die bestehende Hilbert-/Matrixwurzel zu ersetzen. Falls spaeter produktiv benoetigt, duerfen quaternionische Strukturen nur als explizit benannte dark-sektor- oder randgebundene Satz- bzw. Consumeroberflaechen auftreten, nicht als globaler Default. Dazu gehoert auch die komparative Forschungslesart, dass SU(2)-basierte Geometriequantisierung als H-nahe bzw. dunkelseitig isolierte Sektorquantisierung gelesen werden koennte; solche Vergleiche bleiben ausdruecklich Hypothesen- und Diagnoseoberflaechen. Der Cayley-Dickson-Pfad selbst gilt hier bereits als strukturell vorher geschlossene Beweispflicht I und wird nicht erneut als Handoff-Nebenprodukt modelliert. Insbesondere ist theorematisch oder mindestens statusledger-seitig getrennt dokumentiert, dass auf der **H**-Stufe die Matrixalgebra noch assoziativ bleibt, waehrend die tensoriell-kompositionelle Skalarseite an der Nichtkommutativitaet scheitert
S19	A-interner CD-Pfad aus S6b als legaler upstream-Vorgaenger	die dritte Komplementierungsstufe ist als Forschungsseed getrennt sichtbar: Alternativitaet/Nichtassoziativitaet, der Hurwitz-Stop bei 1/2/4/8 und die moegliche CNNA-Lesart eines algebraisch erzwungenen Endes der Komplementhierarchie sind explizit voneinander getrennt dokumentiert. Falls spaeter produktiv benoetigt, duerfen oktonionische Strukturen nur als explizit benannte aeussere Rand- oder dark-sector-Spezialoberflaechen auftreten, nicht als stiller Ersatz des globalen **C**-Pfads. Insbesondere ist statusledger-seitig getrennt markiert, dass auf der **O**-Stufe bereits Matrixmultiplikation nicht mehr assoziativ ist, waehrend exzeptionelle Horizonte wie $J_3(\mathbf{O})$ und die Drei-Generationen-Korrespondenz nur als benannte Fernziele mit Abhaengigkeiten auf **H**-Seed, **O**-Seed, den strukturell geschlossenen CD-Pfad und spaetere DHR-/Matter-Seeds erscheinen; offen bleibt markiert, was externer Satz und was CNNA-spezifische Folgehypothese ist

Phase	Betroffene Module	Muss am Phasenende wahr sein
S20	`PillarA/Structural/BuildAll`, `PillarA/ScalarEmergence/BuildAll`, `PillarA/BuildAll`, `Handoff/Step1MathData`	der erweiterte Strengthening-Pfad schliesst nun architektonisch in Pillar A selbst: `PillarA/ScalarEmergence/BuildAll` existiert, `PillarA/BuildAll` konsumiert `PillarA/Structural/BuildAll` und `PillarA/ScalarEmergence/BuildAll`, und die internen Strukturpfade bleiben dabei strikt pre-handoff. Spaetere Pfeiler erhalten nicht nur den reifen A-Generator, sondern auch eine explizite Scalar-Emergence-Statusoberflaeche, um Spezialregime etablierter Physik als echte Reduktions-, Grenzfall-, Dynamik-, Netz-, Zustandsraum-, Materie- oder AQFT-Fragen zu pruefen; zugleich ist auditiert, dass eine vollstaendige `ScalarClassification` weiterhin Fernziel jenseits des reinen A-Kerns und nicht stillschweigend als erreicht verkauft wird

14.1 Verbindliche Zuordnungstabelle fuer den normbezogenen Teilpfad von Beweispflicht I

Die normbezogenen Teilphasen S2.1–S2.3 des Beweispflicht-I-Pfads sind planlogisch *nicht* im Scalar-Emergence-Block zu beweisen, sondern gehoeren primaer in den strukturellen Cayley-Dickson-Pfad aus S6b. S15–S19 liefern dafuer die Statusdisziplin, die spaetere semantische Lesart und die diagnostischen Folgepfade, aber *nicht* den primaeren Beweisort. Die folgende Tabelle ist deshalb verbindlich zu lesen:

Teilphase	Primaerer Implementationsort	Spaetere Konsum- und Statusorte	Verbindliche Lesart im Plan
S2.1	S6b: `CD/*`	S15: `StatusLedger` S16: `OneSectorRealSeed`	carrier-level `star`, Scalar-Part- und `normSq`-Kandidat sowie die zugehoerige Positivitaetsoberflaeche werden als *struktureller* CD-Beweis vorbereitet. S15 fuehrt nur den Status, S16 liest das Ergebnis spaeter als $\mathbf{R}_{\geq 0}$-artige Einzelsektor-Vorstufe; weder S15 noch S16 ersetzen hier den Beweisort.
S2.2	S6b: `CD/*`	S15: `StatusLedger` S17: `ComplexEmergenceBridge` S18: `QuaternionicSeed`	Multiplikations-, Involutions- und erste Kompositionsvertraeglichkeit des kanonischen Carriers gehoeren weiterhin in den strukturellen CD-Pfad. S17 konsumiert dies spaeter nur als semantische **C**-Constraint-Oberflaeche in der Form einer *ComplexEmergenceBridge*; S18 liest den bereits geschlossenen CD-Pfad als legalen upstream-Vorgaenger fuer die quaternionische Diagnose- und Folgeoberflaeche.

Teilphase	Primaerer Implementationsort	Spaetere Konsum- und Statusorte	Verbindliche Lesart im Plan
S2.3	S6b: `CD/*`	S15: `StatusLedger` S17: `ComplexEmergenceBridge` S19: `OctonionicSeed`, `HurwitzStopSeed`	die Kreuzterm-Analyse von normSq(mul a b) ist die harte Wahrheitsprobe der aktuellen Normrepraesentation. Hier ist exakt zu entscheiden, ob Kreuzterme verschwinden, sich auf kontrollierte Familienbeitraege reduzieren oder eine Architekturerweiterung erzwingen. S17 konsumiert das Ergebnis nur fuer die spaetere Kompositionalitaets-Lesart der operativen **C**-Schicht in der Form einer *ComplexEmergenceBridge*; S19 fuehrt davon getrennt die octonionische Folgeoberflaeche und den Hurwitz-Stop als spaeteren externen bzw. diagnostischen Randstein.

Bindende Schlussregel. S2.1–S2.3 werden damit als normbezogener Teilpfad von Beweispflicht I *primaer in S6b* implementiert. Der Scalar-Emergence-Block ab S15 fuehrt dazu die Statusdisziplin, die spaetere semantische Schichtung und die diagnostischen Folgepfade, konsumiert den strukturell geschlossenen CD-Pfad aber nur als legalen upstream-Vorgaenger. Praezise bleibt dabei die in `MatrixNorms` gefuehrte Frobenius-Norm die operative Vergleichsoberflaeche des `ExecComplex`-Generators ohne ontischen Anspruch, waehrend die ontische Normkandidatin fuer die Skalarstaffelung erst aus der geschlossenen S8-Spektralwurzel als basisinvariante, star-abgeleitete Spektralnorm-Lesart in S16–S19 einfliesst. Insbesondere darf eine spaetere semantische Lesart in S16–S19 nie als Ersatz fuer die strukturelle Kreuzterm- und Kompositionsanalyse in S6b ausgegeben werden.

Fortschreibung von S6b nach gruendem Abschluss. Mit dem gruenden Abschluss von `PillarA/Structural/CayleyDickson/*` gilt Duty I im Sinn der aktuellen formalen Pflichtstruktur als geschlossen: Slot 1, Slot 2 und Slot 3 sind auf dem strukturellen CD-Pfad proof-carrying abgearbeitet. Der verbleibende spaetere Nachzug fuer Slot 2 betrifft damit nicht mehr die formale Schliessung von Beweispflicht I selbst, sondern nur noch die spaetere semantische Nachschaerfung innerhalb des Scalar-Emergence-Blocks. Daraus folgt planlogisch: S6b ist von Beweispflicht I entlastet, `ProofObligationI/*` darf hoechstens noch eine duerre Ledger-/Abschlussoberflaeche tragen, und S6c ist als naechster Arbeitsschritt fuer die gerichtete Handoff-Neustrukturierung freigegeben; Beweispflicht II bleibt strukturell vorbereitet und ab S10a inhaltlich oeffnungsfaehig, Beweispflicht III bleibt die spaetere theorie-erzwungene Rueckwirkung von B/C nach A.

Fortschreibung von S6c nach gruendem Abschluss. Mit dem gruenden Abschluss der gerichteten Handoff-Neustrukturierung ist die bis dahin flache Ablage unter `PillarA/Handoff` semantisch verbindlich aufgeloest: `Core/*` traegt nur noch die internen A-seitigen Kernexporte, `Outputs/*` die ausgehenden Pfeilerkanaele, `Inputs/*` die einzigen spaeter zulaessigen Rueckwirkungskanaele, und `ProofObligationI/II/III/*` die duerre Pflicht- bzw. Vorbereitungsoberflaeche. Parallel dazu gelten nun zwei A-interne Nicht-Handoff-Unterbaeume als gueltige Zielstruktur des Plans: `PillarA/Structural/*` fuer strukturelle Meta-Beweise wie den Cayley-Dickson-Pfad und `PillarA/ScalarEmergence/*` fuer statusgetrennte Prae-Numerik-, Kopplungs- und Folge-Seeds. Keiner dieser beiden Bereiche ist ein zweiter Root-Generator; beide duerfen den gerichteten A-Pfad nur ueber explizit benannte pre-handoff upstream- oder Diagnosebeziehungen konsumieren. Daraus folgt planlogisch: S6c ist abgeschlossen, die Ordnersemantik ist nun bindend festgelegt, und S7a beginnt auf einer bereinigten Kopplungsoberflaeche ohne flache Handoff-Restsemantik.

Fortschreibung von S7a nach gruendem Abschluss. Mit dem gruenden Abschluss der Kopplungsnachschaerfung ist die in P21/P22 geforderte Dualitaet nicht mehr nur planisch festgelegt, sondern

im aktiven Pfad operativ geschlossen: `GeneralizedDtNStrong` traegt raw und stabilized als explizit getrennte, theorematisch gebundene Operatoroberflaechen derselben legalen Coupling-Provenienz, und `MultiSchurStrong` konsumiert fuer Restriktion, Gluing und reduzierte Schur-Komposition ausschliesslich den stabilisierten Pfad. Die Referenz- und Variationsbuilder bleiben dabei familiesicher auf dieselbe Kopplungslogik ausgerichtet; insbesondere erscheint `interfaceInverse` nicht mehr als lose oeffentliche Wurzelgroesse, sondern nur noch als innerhalb von P22 gebundene Interface-Eliminations- bzw. Reduktionslogik. Daraus folgt planlogisch: S7a ist abgeschlossen, die Coupling-Dualpfad-Regel ist im aktiven Pfad nun bindend realisiert, und S7b ist als naechster spaeter Traegerschritt freigegeben.

Fortschreibung von S7b nach gruendem Abschluss. Mit dem gruenden Abschluss der spaeten Traeger-/Kanalnachschaerfung ist die fuer P23 geforderte Trennung von endlicher aktiver Schicht und spaeter exportierbarer Limesoberflaeche im Strengthening-Pfad nun explizit operationalisiert: `InfiniteCarrierStrong` bleibt strikt spaeter Consumer des bereits geschlossenen Netzpfads, exportiert aber kanonische stabile Ausschnitte und gerichtete Uebergaenge weiterhin proof-carrying fuer Vergleichs- und spaetere Directed-Limit-Lesarten; zugleich ist die Referenz-/Variationsoberflaeche bis in diese Stufe hinein familiesicher fortgezogen. Begleitend bleibt `SectorChannelsStrong` ein flankierender Supportconsumer derselben legalen Coupling-Provenienz und eröffnet weder eine neue Netzsemantik noch einen Handoff- oder Closure-Ersatzpfad. Daraus folgt planlogisch: S7b ist abgeschlossen, die spaete P23-Stufe ist fuer den aktiven Pfad hinreichend konsolidiert, und S8a kann nun unmittelbar als erste Unterphase der endlichen Spektralwurzelphase auf dem bereits A-seitig erzeugten Dirichlet-Operator beginnen. Die S8-Phase wird ab jetzt explizit in S8a–S8f aufgefaechert; S9 und die spaeteren Netzwerk-/Diagnoseblöcke bleiben in ihrer bisherigen Reihenfolge unverändert.

Fortschreibung von S8a nach gruendem Abschluss. Mit dem gruenden Abschluss des ersten Spektralschritts ist die fuer den Generator bindende Dualstrangarchitektur nun auch auf der endlichen Spektralwurzel explizit realisiert: `Finite/SpectralDecomposition` bleibt die einzige oeffentliche operative `ExecComplex`-Oberflaeche, `Finite/SpectralDecompositionC` ist als analytischer **C**-Spiegelstrang sichtbar getrennt, und `Finite/SpectralBridge` erzwingt die Konsumrichtung ueber eine explizite injektive Bruecke. Der gruene Abschlussstand zeigt damit bereits, dass der Architekturflip build-stabil ist, ohne die eigentliche allgemeine Spektralhaerte vorzutäuschen. Daraus folgt planlogisch: S8a ist abgeschlossen; S8b beginnt als operative Verschärfung der proof-carrying `ExecComplex`-Oberflaeche, waehrend S8c und S8d den analytischen Spiegel und seine Transferlogik sichtbar und auditierbar nachziehen. S8e und S8f bleiben der explizite Folgeblock fuer einen moeglichst allgemeinen computablen Hermitesch-Pfad und duerfen den bereits gruenden operativen Produktivpfad nicht rueckwirkend in einen zweiten analytischen Generator umdeuten.

Fortschreibung von S8b nach gruendem Abschluss. Mit dem gruenden Abschluss der operativen Nachschaerfung ist die oeffentliche `ExecComplex`-Spektraloberflaeche nun planlogisch als proof-carrying Minimalwurzel geschlossen: kanonische Kandidaten-, Schatten- und Zertifikatsdaten des A-seitig erzeugten endlichen Dirichlet-Operators sind auswertbar vorhanden, erste projektorielle und kommutatorische Zertifikate koennen spaeter proof-carrying konsumiert werden, und der operative Produktivpfad bleibt frei von oeffentlichen `noncomputable`-Definitionen. Der gruene Abschluss hat dabei zugleich eine enge vorbereitende Vorziehung eines kleinen Teils des urspruenglich sichtbaren S8e-`ExecSpectral`-Folgeblocks erzwungen: Nicht der allgemeine computable Hermitesch-Pfad ist bereits geschlossen, wohl aber genau die minimale operative Vorstufe, die fuer eine legal geschlossene proof-carrying Exec-Wurzel noetig war. Daraus folgt planlogisch: S8b ist abgeschlossen; S8c und S8d bilden die sichtbare Folge von auditierbarem Spiegel- und Brueckenausbau; S8e schrumpft auf den verbleibenden allgemeinen computablen `ExecSpectral`-Block, also insbesondere auf allgemeine Polynomarithmetik-, Charakteristikums-, Wurzelisolations-, Kerneigenvektor-, projektorielle und Zertifikatslogik jenseits des bereits gruenden Minimalabschlusses; S8f bleibt die eigentliche Schliessungsphase der allgemeinen algebraischen Spektralwurzel.

Fortschreibung von S8c nach gruendem Abschluss. Mit dem gruenden Abschluss des analytischen Spiegelausbaus ist die fuer den Dualstrang bindende Trennung nun auch auf der hermiteschen **C**-Diagonalisierungsseite proof-carrying geschlossen: `Finite/SpectralDecompositionC` traegt Ei-

genwerte, Eigenvektorbasis, unitaere Diagonalisierung und Spektralsatz des gespiegelt eingebetteten Dirichlet-Operators lokal und auditierbar im markierten analytischen Parallelstrang, waehrend die oeffentliche operative `ExecComplex`-Oberflaeche weiterhin unberuehrt bleibt. Der gruene Abschlussstand zeigt dabei zugleich, dass die mathlib-seitige Spektralmaschinerie nur als enge lokale `noncomputable`-Insel des Spiegelpfads konsumiert wird und dass die Schliessung ueber explizite strukturelle Beweise erfolgt, statt die operative Produktivoberflaeche durch abkuerzende Transferbehauptungen aufzuweichen. Daraus folgt planlogisch: S8c ist abgeschlossen; S8d ist nun der naechste aktive Schritt und muss die Brueckenlogik zwischen operativer `ExecComplex`-Minimalwurzel und analytischem **C**-Spiegel explizit theoremisieren; S8e und S8f bleiben in ihrer nach S8b bereits geschaerften Rolle unveraendert der nachgelagerte sichtbare Folgeblock fuer den verbleibenden allgemeinen computablen Hermitesch-Pfad.

Fortschreibung von S8d nach gruendem Abschluss. Mit dem gruenden Abschluss der Brueckennachschaerfung ist die fuer den Dualstrang bindende Transfersemantik nun auch auf der sichtbaren Uebergangsstufe proof-carrying geschlossen: `Finite/SpectralBridge` bindet die operative `ExecComplex`-Minimalwurzel und den analytischen **C**-Spiegel theorematisch als denselben A-seitig erzeugten endlichen Dirichlet-Operator, traegt Matrixidentifikation und Hermitizitaetstransfer explizit ueber die injektive Bruecke und macht Spur-, Determinanten-/Charakteristikums-, Diagonal- sowie erste projektorielle Schatten nur als kontrollierte Transferklasse konsumierbar. Der gruene Abschlussstand zeigt dabei zugleich, dass die Brueckenstufe keinen zweiten Produktivpfad oeffnet: der analytische Spiegel bleibt auditierbarer Parallelstrang, waehrend spaetere operative Consumer weiterhin ausschliesslich auf der oeffentlichen `ExecComplex`-Oberflaeche aufsetzen muessen. Daraus folgt planlogisch: S8d ist abgeschlossen; S8e verengt sich weiter auf den verbleibenden allgemeinen computablen `ExecSpectral`-Block jenseits der bereits gruenden Minimal- und Brueckenschicht, und S8f bleibt unveraendert die eigentliche Schliessungsphase der allgemeinen algebraischen Spektralwurzel.

Fortschreibung von S8e nach gruendem Abschluss. Mit dem gruenden Abschluss des sichtbaren `ExecSpectral`-Folgeblocks ist die nach S8b und S8d verbleibende computable Restschicht des operativen Spektralpfads nun ebenfalls proof-carrying geschlossen: `PolynomialCore`, `CharpolyCore`, `RootIsolation`, `EigenvectorKernel`, `ProjectorCore` und `Certificates` liefern die allgemeine computable Polynomarithmetik-, Charakteristikums-, Wurzelisolations-, Kerneigenvektor-, projektorielle und Zertifikatsoberflaeche, ohne die bereits gruene Minimalwurzel der oeffentlichen `ExecComplex`-Oberflaeche nachtraeglich umzudeuten oder den analytischen Spiegelstrang als versteckten zweiten Generator zu oeffnen. Der gruene Abschlussstand zeigt dabei zugleich, dass S8e nicht die eigentliche Schliessung der algebraischen Spektralwurzel vorweggenommen hat, sondern genau den noch offenen sichtbaren Restblock build-stabil isoliert und geschlossen hat. Daraus folgt planlogisch: S8e ist abgeschlossen; S8f ist nun der aktive naechste Schritt und hat die operative `SpectralDecomposition`-Oberflaeche ausschliesslich als Consumer des bereits gruenden `ExecSpectral`-Blocks zu einer moeglichst allgemeinen algebraischen Spektralwurzel zusammenzufuehren oder eine verbleibende Restgrenze explizit und plan-konform einzuengen; die Reihenfolge von S9 und den spaeteren Folgephasen bleibt dabei unveraendert.

Fortschreibung von S8f nach gruendem Abschluss. Mit dem gruenden Abschluss der oeffentlichen Spektralschlussphase ist die endliche algebraische Spektralwurzel des aktiven Produktivpfads nun planlogisch insgesamt geschlossen: `Finite/SpectralDecomposition` konsumiert die zuvor sichtbare `ExecSpectral`-Restschicht nun selbst, die proof-carrying Exportoberflaeche fuer die verfuegbaren algebraischen Eigenwert-, Eigenvektor- und Projektorinformationen ist in der oeffentlichen operativen Schicht rueckgebunden, und der analytische **C**-Spiegel bleibt auditierbarer Parallelstrang statt zweiter Generator. Daraus folgt planlogisch streng: Es gibt keine weiteren Unterphasen S8g+, und weder Kesten–McKay-Recovery noch Gap-als-Masse-, RG- oder Norm-/Kompositionsfragen duerfen als semantisch ueberdehnte Fortsetzung in die geschlossene Spektralwurzelphase zurueckgeschoben werden. Der naechste aktive Einstieg bleibt unveraendert S9 fuer endliche Zustands-, Dynamik- und Kanal-Seeds; regimebezogene Spektralobservablen, spaetere Universalitaets- bzw. Recoveryfragen und ihre Statusmarkierung gehoeren in die nachgelagerten Diagnose- und Ledger-Bloecke S10b/S10c

bzw. S15ff. Insbesondere ist festzuhalten, dass S8 zwar die algebraische Spektraloberflaeche des endlichen Dirichlet-Pfads schliesst, aber weder bereits den Kesten-Satz fuer den unendlichen regulaeren Baum noch McKays Konvergenzsatz fuer endliche lokal baumartige regulaere Graphfolgen als CNNA-interne Recovery bewiesen hat. Ebenso bleibt `SpectralGapSeed` spaeter nur ein explizit markierter Masse-/Kopplungskandidat, solange keine eigene Universalitaets- oder Konvergenzaussage bewiesen ist; und eine Levelfolge solcher Groessen wird erst nach expliziter Reskalierungs- und Coarse-Graining-Struktur als RG-Fluss im engen Sinn lesbar.

Fortschreibung von S9 nach gruendem Grundgeruest. Der inzwischen gruende S9-Code belegt, dass die beabsichtigte Leiter `StateSpace` → `MatrixExponential` → {`GibbsStateSeed`, `UnitaryEvolution`} → `DynamicsAdapter` → `ChannelSeed` architektonisch angelegt und in den aktiven Build eingezogen ist. Zugleich zeigt derselbe Codebefund aber praezise, warum S9 im strengen Plansinn noch nicht geschlossen ist: ueber die sechs S9-Dateien liegen derzeit 28 oeffentliche `noncomputable def`; `Finite/MatrixExponential` exponiert oeffentlich noch direkt `NormedSpace.exp`; `Finite/GibbsStateSeed` konsumiert oeffentlich `mirror.eigenvalues` und `mirror.eigenvectorUnitaryMatrix`; `Finite/UnitaryEvoluti` fuehrt Heisenberg- und Schroedinger-Konjugation aktuell noch formelgleich; und `Finite/ChannelSeed` ist bislang nur eine duenne algebraische Package-Schale ohne explizite Kraus-/Choi-/CPTP-Basis. Daraus folgt planlogisch: Das gruende S9-Geruest ist keine abgeschlossene Endphase, sondern der Startpunkt eines expliziten S9x-Haertungs- und Schliessungsblocks S9a–S9h. Diese Auffaecherung zieht die S9-Phase nicht in Richtung von Kesten–McKay-, Gap-/Masse-, RG- oder Norm-Universalitaetsbehauptungen auseinander; sie dient ausschliesslich dazu, die endliche Zustands-, Dynamik- und Kanaloberflaeche ueber der bereits geschlossenen S8-Spektralwurzel auf die im Plan geforderte Form einer oeffentlichen computablen Produktivschicht mit lokal eingehegtem **C**-Spiegel zurueckzufuehren.

Fortschreibung von S9h nach gruendem Abschluss. Mit dem gruenden Abschluss des S9-Haertungs- und Schliessungsblocks ist die endliche Zustands-, Dynamik- und Kanaloberflaeche des aktiven Produktivpfads nun planlogisch insgesamt geschlossen: `StateSpace`, `MatrixExponential`, `GibbsStateSeed`, `UnitaryEvolution`, `DynamicsAdapter` und `ChannelSeed` bilden im gruenden Stand eine oeffentliche computable ExecComplex-Produktivschicht mit nur lokalem und explizitem **C**-Spiegel. Der Abschlussbefund bedeutet dabei gerade keine inhaltliche Ausweitung der spaeteren Folgephasen, sondern eine Statusschaerfung: S10a bleibt unveraendert der naechste aktive Einstieg, S10b und S10c behalten ihre Rolle als nachgelagerte Diagnose-, Regime- und Ledgerbloecke, S11a/S11b bleiben in ihrer bisherigen Reihenfolge erhalten, und der A-interne Scalar-Emergence-Block S15–S20 wird durch S9h nicht vorgezogen oder semantisch aufgeladen. Gleiches gilt fuer den nach gruener A6-Schliessung separat gefuehrten Arithmetik-Folgeblock gemaess Abschnitt 15: er ist als eigener A-interner Folgepfad explizit angeschlossen, ersetzt aber weder S10a noch Beweispflicht II. Neu ist nur, dass Beweispflicht II ab dem unveraendert vorgesehenen Einstieg S10a nun nicht mehr unter einem offenen S9-Vorbehalt steht.

15 A-interner Arithmetik-Folgeblock A1–A21d auf konsolidiertem gruenden A11-Stand

Einordnung. Der unter `PillarA/Arithmetic/*` im letzten konsolidierten gruenden A11-Stand erreichte Arithmetikpfad A1–A11 ist ein eigener A-interner Folgepfad ueber der bereits geschlossenen endlichen Spektralwurzel; er ist weder zweiter Root-Generator noch Handoff-Ersatz und darf insbesondere weder `Outputs/A_to_*` noch `ABHandoffStrong` noch `PillarAStep1Closed` als internen Abkuerzungsweg konsumieren. Der Arithmetikpfad bleibt damit strikt pre-handoff, streng derived-only ueber der A-seitigen Spektral- und Formenoberflaeche und architektonisch von Beweispflicht II getrennt. Neu bindend ist jedoch seine interne Re-Ankerung: nicht CM-Punkt, Gitter oder spaetere modulare Invarianten bilden seinen Ursprung, sondern die in `Finite/SpectralDecompositionCore`, `Finite/SpectralDecompositionC` und `Finite/SpectralBridge` bereits angelegte Dualwurzel aus oeffentlich operativem `ExecComplex`-Strang, explizit separatem **C**-Spiegel und bewiesener Brueckenlage. Seine Aufgabe ist nicht, S10a zu ersetzen, sondern die nach gruener A6-Schliessung sichtbar

gewordene quadratische Formen-, Ordnungs- und modulare Anschlussseite des Projekts eigenstaendig, aus genau dieser Spektralwurzel heraus und statusdiszipliniert zu schliessen.

Revidierter Befund zum gruenden Ausgangspunkt A1–A6. Der gruende A6-Stand ist keine homogen vollgeschlossene A1–A6-Kette, sondern eine ungleich starke Anfangsschicht. Formal vorhanden sind `MobiusTransfer`, `ArithmeticSignature`, `ReducedQuadraticForm`, `RingTypeRecognizer`, `QuadraticFactorWitness`, `QuadraticFactorCertificate`, `BinaryQuadraticFormAction`, `ReducedFormTheory`, `QuadraticOrderRecognizer`, `GaussianSeed` und `FiniteReducedEnumeration`; mathematisch sind diese Module jedoch verschieden weit geschlossen. Zugleich zeigt die gruende A11-Arbeitskopie rueckblickend, dass der Arithmetikpfad bereits faktisch auf der Spektral-Dualwurzel ruht: `MobiusTransfer` importiert `Finite/SpectralDecomposition`, `ArithmeticSignature` konsumiert die operative Spektralseite, und die spaeteren CM-/Gittermodule bauen nur downstream darauf auf. A2 ist in der aktuellen Repo-Lesart bereits substantiell tragfaehig, waehrend A1, A3, A4, A5 und A6 noch je eigene Verstaerkungsarbeit benoetigen. Insbesondere ist A1 bislang nur als punktweise `EvalPolynomial`-Faktorisierungsoberflaeche und nicht als interne Polynomteilbarkeit geschlossen; A3 traegt Aequivalenz, Diskriminanteninvarianz und elementare T/S-Zuege, aber noch keinen allgemeinen Reduktionssatz; A4 liefert erst einen minimalen Deskriptor fuer Orientierung und Ringtyp, noch ohne saubere Trennung von Rohdiskriminante, Ordnungsdiskriminante, fundamentaler Felddiskriminante und Leiter; A5 bleibt ein konditionaler Gauß-Seed ueber einer vorgegebenen Gewichtsrelation und gewinnt noch kein echtes Zertifikat direkt aus dem operativen Signaturpfad; A6 liefert eine kleine bewiesene Fallbibliothek reduzierter Formen, aber noch keine Vollstaendigkeit der Enumeration und noch keine kleine Klassensemantik.

Status der konsolidierten Vorstufe A7–A11 im gruenden A11-Stand. Ueber die gehaertete Anfangsschicht hinaus sind im konsolidierten A11-Stand bereits eine erste Diskriminanten-/Ordnungslesart, die kleine reduzierte Klassenwelt, der modulare Bridge-Layer sowie die theorematische CM-/Gittervorstufe vorhanden. Diese spaeteren Module werden im vorliegenden Plan jedoch bewusst nicht rueckwirkend als bereits vollgeschlossener Endblock gelesen: A7–A9 tragen eine kleine, noch auf die aktuell bewiesene Welt beschraenkte Klassen- und Ordnungsseite; A10 schliesst die Formenwelt an die offizielle modulare Toolchain an; A11 schliesst den proof-carrying Vorraum von CM-Punkt und Periodengitter, jedoch noch ohne oeffentliche Weierstraß-/j-Schicht. Genau deshalb bleibt die Reihenfolge A11 → A11b → A12a–A12k bindend: erst die Strukturphase zieht die Repo-Oberflaeche in thematische Unterordner, erst danach beginnt der eigentliche Invariantenlift.

Bindende Architekturregel. Der Arithmetikblock ist damit nicht erst ab A7 ernstzunehmen, sondern bereits vom Anfang her strikt re-anchored zu staffeln. Die bindende Kernkette lautet nun explizit: endliche bright-DtN-Spektralwurzel mit operativem `ExecComplex`-Strang, **C**-Spiegel und Bruecke → operativer Arithmetikstrang (A1–A8) → semantische Vorstufe ohne ersten echten Lift (A9–A11) → strukturelle Umordnung der Arithmetikmodule nach Spezialisierung innerhalb des Arithmetic-Pfads (A11b) → erster modulare Invarianten- und gegebenenfalls lokaler `noncomputable`-Lift (A12a–A12k) → kleine Klassen-, Extraktions- und Rueckkopplungsconsumer (A13–A19) → Fernhorizonte A20/A21a–A21d. Daraus folgen vier harte Ordnungsregeln. Erstens: die gruene Anfangsschicht A1–A6 darf im Plan nicht laenger staerker gelesen werden, als der Code sie derzeit traegt, und ist primaer als Consumer der Spektralwurzel zu lesen, nicht als Vorstufe einer bereits fertigen CM-Welt. Zweitens: die fehlenden Schliessungen werden als eigener Strengthening-Block A1b–A6b eingefuehrt, der die existierenden A1–A6-Module jeweils gezielt haertet, bevor die weiterfuehrende Folgearbeit A7–A19 den Pfad globalisiert. Drittens: A9–A11 duerfen semantische CM- und Gitterobjekte einfuehren, bleiben aber planlogisch noch vor dem ersten eigentlichen `noncomputable`-Lift; A11b zieht danach die bis dahin flach nebeneinanderliegenden Arithmetikmodule in thematische Unterordner innerhalb von `PillarA/Arithmetic/*` um, ohne neue mathematische Inhalte zu behaupten oder die bis A11 geschlossene Beweislage umzudeuten. Weder duerfen A9–A11 bereits A12-Invarianten vorwegnehmen noch darf A11b semantisch als neuer Inhaltsschritt statt als strukturierende Reorganisation gelesen werden. Viertens: A13 steht bewusst vor A14, weil zuerst die kleine semantische Ziel-API ueber A12j/A12k stabilisiert werden muss; die echte Rueckeinspeisung konkreter CNNA-Signaturen in diese

bereits geschlossene kleine Zielwelt erfolgt erst in A14–A17. Die Reihenfolge ist daher als (A1 → A1b), (A2 → A2b), (A3 → A3b), (A4 → A4b), (A5 → A5b), (A6 → A6b), und danach erst als (A1b, A2b, A3b, A4b, A5b, A6b) → A7 → A8 → A9 → A9b → A10 → A11 → A11b → A12a → A12b → A12c → A12d → A12e → A12f → A12g → A12h → A12i → A12j → A12k → A13 → A14 → A15 → A16 → A17 → A18 → A19 → A20 → A21a → A21b → A21c → A21d zu lesen. Insbesondere duerfen A7–A19 die offenen Luecken von A1, A3, A4, A5 und A6 nicht stillschweigend ueberspringen; ebenso duerfen A12–A19 die operative Spektralwurzel nie semantisch ersetzen, sondern nur kontrolliert konsumieren.

Praezisierung des modularen Spaetblocks nach TomS #30. TomS hebt im letzten Thread-Beitrag ausdruecklich hervor: „An der Stelle moechte ich noch auf eine Beziehung zu den eingangs diskutierten speziellen Gittern ueber den komplexen Zahlen hinweisen.“ Danach fuehrt er den Pfad von einem Punkt τ der oberen Halbebene ueber das komplexe Gitter $\Lambda_\tau = \mathbf{Z} + \tau\mathbf{Z}$, Eisensteinreihen und die Weierstraß-Invarianten g_2, g_3 bis zur j-Invariante aus; Quelle: TomS, Beitrag #30 im Thread *„sich durch die Mathematik treiben lassen …“* auf astronews, https://www.astronews.com/community/threads/sich-durch-die-mathematik-treiben-lassen.12586/post-154315. Der Plan liest dies nicht als Vorziehung von A10–A13, wohl aber als bindende inhaltliche Praezisierung dieses spaeteren Blocks: Zwischen CM-Punkt-Seite und j-Invariantenseite darf kuenftig keine rohe semantische Abkuerzung mehr stehen; die Gitter- und Weierstraß-Zwischenlage ist als eigener proof-carrying Schritt mitzuschliessen.

Bindende Dualstrang-Praezisierung ab A11. Ab A11 ist die im Leitsatz bereits global festgelegte Trennung aus `Computability-Budget-Prinzip` und `Parallel-Dualstrang-Prinzip` im Arithmetikblock lokal hart zu machen. Der gruende A11-Stand bleibt dabei noch explizit *vor* dem ersten eigentlichen Invarianten- und `noncomputable`-Lift: `CMPointShadow`, `CMPoint` und `PeriodLattice` sind als theorematische Vorstufe ueber derselben Spektralwurzel zu lesen, nicht bereits als oeffentlicher modularer Endblock. Erst ab A12a darf – falls die gepinnte Toolchain die semantische Seite nicht selbst computabel traegt – ein unvermeidlicher lokaler `noncomputable`-Spiegelstrang oeffentlich in die Planung eintreten. Ab dort ist jede CM-, Gitter-, Weierstraß- oder j-seitige Oberflaeche explizit in einen oeffentlich computablen Shadow-/Zeugenstrang und einen davon getrennten semantischen Spiegelstrang zu staffeln. Beide Straenge duerfen nur ueber bewiesene Bruecken, Transport- oder Existenzsaetze gekoppelt werden. Der semantische Spiegelstrang darf den operativen Produktivpfad nie ersetzen; umgekehrt ist ein computabler Schatten nie als semantische Gleichsetzung mit dem abstrakten Invariantenobjekt zu lesen. Diese Trennung ist nicht nur fuer A11, die anschliessende Strukturphase A11b und den nun in A12a–A12k aufgefaecherten Invariantenblock, sondern fuer alle folgenden arithmetischen Consumer A13–A21d bindend mitzuschleppen.

Bindende Nachschaerfung nach TomS #42, #43, #49 und #53. Der spaetere Thread praezisiert vier Punkte, die planlogisch nicht mehr implizit bleiben duerfen. Erstens stellt TomS zu elliptischen Kurven klar, dass diese „erst separat zu konstruieren“ seien und dass die Kompaktifizierung ueber einem Gitter zwar in die richtige Richtung fuehre, das Gitter allein aber noch nicht entscheidend sei; Quelle: TomS, Beitrag #42 im selben Thread, https://www.astronews.com/community/threads/sich-durch-die-mathematik-treiben-lassen.12586/page-3. Zweitens markiert er die endlichen einfachen Gruppen ausdruecklich als *neues* viertes Gebiet und damit als von der bisherigen Zahlentheorie-/Funktionentheorie-/Geometrie-Kette getrennten Horizont; Quelle: TomS, Beitrag #43, ebenda. Drittens verschaerft er fuer die modulare Seite, dass es inhaltlich um Modulformen bzw. Modulfunktionen und um ζ-/L-Funktionen gehe, *nicht* speziell um die Riemannsche ζ-Funktion; ausserdem koenne die Gitter-/Zahlkoerperseite in Ausnahmefaellen durch etwas Allgemeineres ersetzt werden, waehrend die modulare Transformationsstruktur in gewisser Weise erhalten bleibe; Quelle: TomS, Beitrag #49, ebenda. Viertens beginnt TomS mit Beitrag #53 die endliche-Gruppen-Seite selbst weiter auszuwickeln: nach der zyklischen Klasse tritt nun die Permutationsseite explizit hervor, mit symmetrischen und alternierenden Gruppen, Erzeugung gerader Permutationen durch 3-Zyklen und der Ausnahme $A(4)$ als nicht-einfacher Sonderfall; Hintergrundverweis: TomS, Beitrag #53, https://www.astronews.com/community/threads/sich-durch-die-mathematik-t

reiben-lassen.12586/post-154366. Der Plan zieht daraus fuenf bindende Konsequenzen: (i) A12 darf j nicht nur als statisches Endwertdatum, sondern muss anschliessend auch als modulare Funktionsseite explizit sichtbar machen; (ii) die Klassenzahlseite A7–A9 ist um die Luecke irreduzibel versus prim bzw. um eine kleine Ideal-/Primeideal-Lesart zu schaerfen, ohne schon volle allgemeine Dedekind-Theorie zu behaupten; (iii) die zetafunktionale bzw. determinantielle Leserichtung wird erst *nach* A19 als eigener Fernblock markiert; (iv) die von TomS angesprochene endliche-Gruppen-Seite wird als expliziter, aber nicht voreilig ueberdehnter Spaethorizont reserviert; und (v) dieser Spaethorizont ist nicht mehr als ein einziger pauschaler Gruppenblock zu lesen, sondern mindestens in zyklische, alternierende/symmetrische, Lie-Typ- und sporadische Folgefenster zu staffeln. Keine dieser Praezisierungen zieht A14 oder spaetere Pillars vor; sie schaerfen allein den Folgevertrag des Arithmetikpfads.

Einheitliche Tabelle der A-Phasen A1–A21d.

Pfadkonvention der Tabelle. Die in der folgenden Tabelle genannten Modulpfade folgen bereits der durch A11b verbindlich gemachten Zielordnerstruktur nach Themenlagen. Fuer Phasen vor A11b ist dies daher als planlogische Zielbenennung derselben Module zu lesen, nicht als Behauptung, dass die historische Vor-A11b-Arbeitskopie diese Verzeichnisstruktur bereits vollzogen hatte.

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
Ausgangsschicht A1–A6 im gruenden Stand			
A1	Ausgang	`Core/QuadraticFactorWitness`, `Core/QuadraticFactorCertificate`	die bisherige Witness-Schicht ist gegenueber dem alten Drei-Punkte-Schatten sichtbar verstaerkt: `QuadraticFactorCertificate` traegt Faktor, Quotient und eine punktweise Faktorisierungsoberflaeche auf der aktuellen `EvalPolynomial`-Lesart. Geschlossen ist damit eine proof-carrying Evaluationsfaktorisierung; noch *nicht* geschlossen ist eine interne Polynomring-Teilbarkeitslesart samt kanonischer Quotientenextraktion aus dem operativen CNNA-Pfad
A2	Ausgang	`Forms/BinaryQuadraticFormAction`	die $SL_2(\mathbf{Z})$-Wirkung auf binaere quadratische Formen ist substantiell vorhanden: `SL2Z`, Generatoren, Substitutionswirkung und Diskriminanteninvarianz sind formal getragen. Die Konventionsseite der rechten Substitutionslesart ist explizit zu respektieren; fuer den aktuellen Stand ist A2 damit der tragfaehigste Block der Anfangsschicht
A3	Ausgang	`Forms/ReducedFormTheory`	Proper-Äquivalenz, Inversen in `SL2Z`, Diskriminanteninvarianz unter Aequivalenz sowie elementare T/S-Reduktionsschritte sind angelegt. Noch nicht geschlossen sind ein allgemeiner Reduktionsalgorithmus, ein Existenzsatz reduzierter Repräsentanten fuer positive definite Formen und eine terminierende Normalformlesart
A4	Ausgang	`Orders/RingTypeRecognizer`, `Orders/QuadraticOrderRecognizer`	der aktuelle Stand erkennt sichere Basisfaelle und trennt die Diskriminantenorientierung deskriptiv in imaginär, reell und degenerat. Geschlossen ist damit eine minimale Ringtyp-/Orientierungsoberflaeche; noch offen bleiben fundamentale Diskriminanten, Leiter, echte Ordnungsdaten und die saubere Rueckbindung von Formklasse zu Ordnung
A5	Ausgang	`Core/GaussianSeed`	ein erster zweiter Seed jenseits des Eisenstein-nahen Einstiegs ist als konditionale Gauß-Schicht vorhanden: aus einer vorgegebenen Gewichtsrelation wird ein Gauß-artiger Witness erzeugt. Noch nicht geschlossen sind eine direkte Ableitung aus `ArithmeticSignature` bzw. der Schur-/Charpoly-Seite und der Sprung vom bloßen Witness zu einem echten Faktorzertifikat

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A6	Ausgang	`Forms/FiniteReducedEnumeration`	fuer eine kleine endliche Welt ausgewaehlter negativer Diskriminanten liegen explizite reduzierte Formen mit Beweisen fuer Reduziertheit und Diskriminantenwert vor. Geschlossen ist damit eine kleine bewiesene Fallbibliothek; noch nicht geschlossen sind Vollstaendigkeit, Dublettenfreiheit als globale Aussage ueber die getragene Liste und eine daraus gewonnene kleine Klassensemantik
Strengthening-Block A1b–A6b fuer die offenen Anfangsluecken			
A1b	Strengthening	`Core/QuadraticFactorCertificateStrong`, begleitend `Core/QuadraticFactorCertificate`	die Zertifikatsseite ist ueber die aktuelle punktweise Evaluationsform hinaus gehaertet: Quotienten- und Faktordaten tragen nun eine kanonische Divisibilitaetslesart auf der aktiven Polynomoberflaeche, und es gibt eine explizite Transportschicht vom Witness-/Signaturpfad zu einem echten Zertifikatstyp. Damit ist A1 nicht mehr nur Interpolationsschatten, sondern eine belastbare algebraische Zwischenstufe fuer spaetere Extraktion
A2b	Strengthening	`Forms/BinaryQuadraticFormOrbit`, begleitend `Forms/BinaryQuadraticFormAction`	die Aktionsseite ist als spaeter konsumierbare Orbit- und Generatoroberflaeche gebuendelt: die Substitutionskonvention ist theorematisch festgezogen, Orbit- und Transportsaetze fuer die T/S-Generatoren sind explizit exponiert, und Proper-Äquivalenz kann ab hier ueber diese API statt ueber rohe Formelauswertung konsumiert werden. A2b ersetzt A2 nicht, sondern macht die bereits grune Aktionsseite downstream-stabil fuer A3 und A10
A3b	Strengthening	`Forms/ReducedRepresentative`, begleitend `Forms/ReducedFormTheory`	die Reduktionsseite ist vom blossen Geruest zum eigentlichen Werkzeug gehaertet: fuer positive definite Formen gibt es einen expliziten Reduktionspfad zu einem reduzierten Repräsentanten, inklusive einer terminierenden oder wohlfundierten Masslesart und der dazugehoerigen Invarianzsaetze. Spaetere Enumerations- und Klassenschritte duerfen ab dann auf einen echten Reduktionssatz bauen und nicht nur auf kuratierte Listen
A4b	Strengthening	`Orders/QuadraticOrderDescriptor`, begleitend `Orders/QuadraticOrderRecognizer`	der bisher minimale Ringtyp-Deskriptor ist zu einer belastbaren lokalen Ordnungsoberflaeche verstaerkt: Rohdiskriminante, Vorzeichenregime und Nichtdegenerationsbedingungen sind explizit getrennt, und die Recognizer-Schicht traegt keine stillschweigende Gleichsetzung von Formdiskriminante, Ordnungsdiskriminante und Felddiskriminante mehr. A4b bereitet damit A7 vor, ohne dessen globale Leiter-/Fundamentaldisziplin vorwegzunehmen
A5b	Strengthening	`Core/GaussianSeedCertificate`, begleitend `Core/GaussianSeed`	der Gauß-Pfad ist von einem konditionalen Witness zu einer echten proof-carrying Seed-Stufe gehaertet: mindestens ein Gauß-artiger Fall wird aus legalen `ArithmeticSignature`- oder Schur-Daten an ein Faktorzertifikat angeschlossen, und der Seed-Namenpfad wird nicht laenger als Ersatz fuer eine tatsaechliche CNNA-Ableitung gelesen. A5b ist damit die lokale Vorstufe fuer die spaetere generische Extraktion in A14
A6b	Strengthening	`Forms/FiniteReducedEnumerationAudit`, begleitend `Forms/FiniteReducedEnumeration`	die kleine A6-Welt ist als saubere endliche Katalogschicht gehaertet: die getragenen Listen sind theorematisch auf ihre interne Konsistenz, Dublettenfreiheit, Descriptor-Kompatibilitaet und explizite Traegerwelt hin auditiert. Noch nicht gefordert ist hier bereits die volle globale Vollstaendigkeit; A6b macht aber exakt sichtbar, was die kleine aktuelle Enumerationswelt behauptet und was erst A8/A9 darueber hinaus schliessen duerfen
Weiterfuehrender Folgeblock A7–A21d nach gehaerteter Anfangsschicht			

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A7	Folgeblock	`Orders/FundamentalDiscriminant`, `Orders/QuadraticOrderData`, begleitend `Orders/QuadraticOrderRecognizer`, begleitend `Orders/QuadraticOrderDescription`	die quadratische Ordnungsseite ist von der bisherigen Rohdiskriminantenlesart geloest: fuer jede relevante negative Diskriminante wird explizit zwischen Ordnungsdiskriminante, fundamentaler Felddiskriminante und Leiter f unterschieden, und `QuadraticOrderData` traegt diese Provenienz proof-carrying. Weder D noch Δ duerfen im Plan noch roh synonym verwendet werden; Gauß- und Eisenstein-Fall erscheinen als Spezialfaelle dieser Datenstruktur und nicht als semantische Kurzschluesse aus dem Seed-Namen
A8	Folgeblock	`Forms/ReducedEnumerationComplete`, begleitend `Forms/FiniteReducedEnumeration`, `Forms/ReducedRepresentative`	die kleine reduzierte Enumeration aus A6/A6b ist nicht mehr nur build-stabil, sondern fuer die aktuell getragenen negativen Diskriminanten vollstaendig und eindeutig: jede reduzierte Form der ausgewaehlten kleinen Diskriminanten liegt in der Liste, die Liste enthaelt keine Dubletten, und jede Proper-Äquivalenzklasse besitzt genau einen reduzierten Repräsentanten in dieser endlichen Welt. Die Beweise laufen explizit ueber die Reduktionsbedingungen und die in A3b geschlossene Reduktionsseite und nicht nur ueber brute-force-Entscheider
A9	Folgeblock	`Orders/ProperClassSet`, optional `Orders/ClassCompositionSmall`	fuer die kleine A6-Welt steht nun eine echte Klassensemantik: Proper-Äquivalenzklassen ausgewaehlter negativer Diskriminanten sind als endlicher Typ oder Quotient formalisiert, ihre Kardinalitaeten sind fuer die getragenen Faelle proof-carrying bestimmt, und mindestens der erste nichttriviale Mehrklassenfall ist explizit isoliert. Zugleich wird hier bereits semantisch vorbereitet, dass die Klassenzahl nicht bloss eine Zaehlgroesse bleibt, sondern spaeter die erste kontrollierte Lesart dafuer traegt, wann Faktorisierung, Klassenstruktur und Primelement-/Idealverhalten auseinanderlaufen. Falls eine kleine Kompositionsoberflaeche eingezogen wird, bleibt sie auf die aktuell tragenden Diskriminanten beschraenkt und ersetzt keine spaetere allgemeine Kompositionstheorie
A9b	Folgeblock	`Orders/PrimeIrreducibleGapSmall`, optional `Orders/PrimeIdealFactorizationSmall`, begleitend `Orders/QuadraticOrderData`	die kleine Ordnungs- und Klassenwelt ist nun in Richtung TomS's Primelement-/Irreduzibilitaetsfrage nachgeschaerft: fuer die aktuell getragenen imaginär-quadratischen Ordnungen wird explizit getrennt, was als irreduzibles Element, als Primelement und – wo die Repo-Lage dies bereits traegt – als Primeideal zu lesen ist. Mindestens die Klassenzahl-Eins-Faelle und der erste echte Nicht-UFD-/Mehrklassenfall sind so gegeneinander gestellt, dass aus der Klassenstruktur keine stille Elementfaktorisierung mehr hineinphantasiert wird; eine volle allgemeine Dedekind-/Idealfaktorisierungstheorie bleibt dabei bewusst spaeteren Verallgemeinerungen vorbehalten

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A10	Folgeblock	`Modularity/ModularBridge`, flankierend offizielle Modular-Objekte der Toolchain	die bisherige eigene $SL_2(\mathbf{Z})$- und Formenaktionsschicht ist auf Basis von A2/A2b an die offizielle modulare Gruppen- und obere-Halbebenen-Welt der Toolchain angeschlossen, ohne die interne Formenoberflaeche zu entsorgen. Eigene `SL2Z`-Zeugen, Proper-Äquivalenz und Diskriminanteninvarianz lassen sich in den offiziellen modularen Rahmen transportieren; der Bridge-Layer bleibt dabei strukturell duenn und eroeffnet keinen zweiten arithmetischen Begriffspfad neben der bereits geschlossenen internen Formenoberflaeche. Zugleich wird hier toolchain-diszipliniert festgehalten, dass die in A11/A11b/A12 benoetigten API-Namen erst gegen die gepinnte Repo-Lage und nicht gegen spaetere Dokumentationsstaende zu finalisieren sind
A11	Folgeblock	`CM/CMPointShadow`, `CM/CMPoint`, begleitend `CM/PeriodLattice`	zu jeder positiven definiten reduzierten Form mit negativer Diskriminante ist eine computability-diszipliniert CM-Oberflaeche als semantische Vorstufe geschlossen: oeffentlich computabel steht ein proof-carrying Shadow-/Zeugenstrang der CM-Daten, und davon getrennt ein semantischer CM-Punkt der oberen Halbebene samt bewiesener Bruecke zwischen beiden. Daraus ist zugleich das zugehoerige komplexe Gitter $\Lambda_\tau = \mathbf{Z} + \tau\mathbf{Z}$ als eigene semantische Zwischenlage sichtbar gemacht; ein computabler Gitterschatten darf hoechstens klar markiert danebenstehen. Proper-Äquivalente Formen liefern modular aequivalente semantische CM-Punkte und kompatible Gitterdaten; fuer reduzierte Formen ist mindestens eine kontrollierte Existenz- oder Transportbeziehung zur Standard-Fundamentaldomaene sichtbar gemacht, ohne bereits den staerkeren kanonischen Direktliegesatz zu erzwingen. Im gruenden A11-Stand bleibt dieser Schritt noch vor dem ersten eigentlichen Invarianten- und `noncomputable`-Lift: Shadow, semantischer Punkt und Gitter sind als theorematische Vorstufe ueber derselben Spektralwurzel zu lesen und nicht bereits als oeffentliche Weierstraß/j-Schicht. Damit schliesst A11 die Luecke zwischen Formen-, modularer Orbit- und Gitterseite, ohne bereits j- oder Klassenkoerperaussagen vorwegzunehmen und ohne den operativen Produktivpfad schleichend `noncomputable` werden zu lassen

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
		`Core/*`, `Forms/*`, `Orders/*`, `CM/*`, `Modularity/*`, `Shadow/*`, `Examples/*`, `Horizons/*`, begleitend `BuildAll`,	
A11b	Folgeblock	`Notation`	der bis A11 inhaltlich geschlossene Arithmetikpfad ist nun auch repo-architektonisch geordnet: die bislang flach nebeneinanderliegenden Module unter `PillarA/Arithmetic/*` werden nach Spezialisierung und Thema in eigene Unterordner innerhalb des Arithmetikpfads umgezogen, ohne dabei die mathematische Ableitungsrichtung zu veraendern. Mindestens die Kernschichten `Core` (spektral gebundene Basisschnittstellen und Signaturen), `Forms` (quadratische Formen, Aktionen, Reduktion, Enumeration), `Orders` (Ringtyp-, Diskriminanten- und Ordnungsdaten), `CM` (CMPointShadow, CMPoint, PeriodLattice und zugehoerige Provenienzvorstufen), `Modularity` (Bridges und spaetere modulare Invariantenmodule), `Shadow` (explizit computable Reihen-, Fourier- und Approximationsoberflaechen), `Examples` (gruen geprüfte End-to-End-Beispiele) sowie `Horizons` (explizit nachgelagerte Zeta-/Gruppen-Fernbloecke) sind strukturell zu trennen; Hilfs- und Zertifikatsmodule duerfen dabei thematisch den tragenden Schichten zugeordnet werden, statt weiter als flache Dateimenge zu verbleiben. `BuildAll`, `Notation` und Importketten sind auf die neue Ordnerstruktur mitzuziehen, sodass Consumer-Disziplin, kurze Importpfade und die Shadow/Spiegel-Trennung auch auf Verzeichnisebene sichtbar werden. A11b ist ausdruecklich eine Struktur- und Hygienephase innerhalb des bereits geschlossenen A11-Stands, kein neuer mathematischer Inhaltsblock und kein Vorgriff auf A12-Invarianten oder spaetere Handoffs
A12a	Folgeblock	`CM/SemanticCMProvenance`, begleitend `CM/CMPoint`, begleitend `CM/PeriodLattice`, begleitend `BuildAll`	die oeffentliche A12-Oberflaeche ist hart auf eine Provenienz-API umgeschnitten und markiert damit erstmals den eigentlichen semantischen Invariantenlift ueber der in A11 bereits geschlossenen CM-/Gittervorstufe: in oeffentlich exportierten A12-Signaturen treten freie semantische Eingaben wie `weightFour`, `weightSix` oder ein freier `j`-Wert nicht mehr als Primaerdaten auf. Stattdessen beginnt jede semantische A12-Konstruktion bei einer expliziten Provenienzquelle; rohe Paketierer, frei normalisierte `of*Weights`-Konstruktoren und analoge Hilfsoberflaechen sind aus `BuildAll`/Notation entfernt oder sichtbar als intern/provisorisch markiert. Falls ein unvermeidlicher lokaler `noncomputable`-Schritt noetig wird, darf er ab hier und nicht schon rueckwirkend in A11 verbucht werden
A12b	Folgeblock	`CM/SemanticCMProvenance`, begleitend `CM/CMPoint`, begleitend `CM/PeriodLattice`	der Provenance-Kern ist als gerichtete Herkunftskette geschlossen: eine Struktur vom Typ `SemanticCMProvenance` traegt Repraesentantenwahl, reduzierte Form und den dazugehoerigen semantischen CM-Punkt, waehrend das zugehoerige Gitter nur noch definitional aus diesem Punkt gewonnen wird. Die offiziellen Ankerfaelle fuer Gauß und Eisenstein sind als bindende Repraesentanten festgelegt; die Gitterseite kann damit nicht mehr getrennt von ihrer CM-Herkunft eingespeist oder spaeter stillschweigend ausgetauscht werden

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A12c	Folgeblock	`CM/SemanticCMProvenance`, begleitend `Modularity/`, `WeierstrassInvariant`	die semantischen Eisenstein-/Weierstraß-Gewichte sind oeffentlich nicht mehr als gespeicherte Rohfelder vorhanden, sondern als aus der Orbitklasse der Provenienz definierte Normalisierungsdaten. Dabei ist fuer den gesamten A12-Block nun explizit genau *eine* globale Normierungskonvention festzunageln – entweder die G_4/G_6-Lesart der Gittersummen oder die normalisierte E_4/E_6-Lesart der q-Reihen – und diese Wahl ist in Signaturen, Notation und Modulnamen offen mitzufuehren. Falls Shadow- und Semantikschicht unterschiedliche historische Konventionen erben, ist ein eigener proof-carrying Konversionssatz zwischen beiden Lesarten verpflichtend einzuziehen; implizites Umdeuten von Koeffizienten oder stilles Mischen der Konventionen ist unzulaessig. A12-public ist hier bewusst auf die aktuell wirklich proof-carrying getragenen CM-Ankerorbits begrenzt; insbesondere wird kein scheinallgemeines `PeriodLattice` $\mapsto$ semantisches Gewichtsobjekt vorgetaeuscht, solange dafuer im Repo noch kein eigener vollstaendiger Brueckensatz formalisiert ist
A12d	Folgeblock	`Modularity/WeierstrassInvariant`, begleitend `CM/SemanticCMProvenance`	die Weierstraß-Schicht ist auf compute-from-source umgestellt: oeffentlich gespeichert wird nur noch Provenienz, und g_2, g_3, Δ werden definitional aus den abgeleiteten Gewichten erzeugt statt als freie Invarianten mittransportiert. Genau an dieser Stelle ist die in A12c fixierte Normierung durch explizite Definitionsgleichungen bzw. einen benannten Umrechnungssatz bis zur Weierstraß-Ebene durchzutragen, damit die Faktoren zwischen E_4/E_6, G_4/G_6 und g_2, g_3 nicht nur proseartig, sondern als sichtbarer Teil der API kontrolliert bleiben. Nichtsingularitaet und die klassischen Gauß/Eisenstein-Ankerwerte sind als Folgesaetze dieser Definitionskette geschlossen und nicht mehr Resultat frei gepackter Rohdaten
A12e	Folgeblock	`Modularity/JInvariant`, begleitend `Modularity/WeierstrassInvariant`, begleitend `Shadow/JSeriesApprox`	die j-Lesart ist oeffentlich nur noch semantische Auswertung von `DerivedWeierstrass`: ein freier `value`-Input oder ein frei konstruierbarer `pack`-Primaerkonstruktor ist aus der public-API entfernt. Ein gegebenenfalls nur lokal `noncomputable`es j ist damit sauber vom computablen Schatten getrennt und bleibt vollstaendig an die Provenienz- und Weierstraß-Ableitung rueckgebunden
A12f	Folgeblock	`CM/SemanticCMProvenance`, `Modularity/JInvariant`, begleitend `CM/CMPoint`	der positive Eisenstein-Repraesentant ist nicht mehr als eigener semantischer Zweig modelliert, sondern nur noch als Transport desselben Eisenstein-Orbits. Daraus folgen Gleichheit der abgeleiteten Gewichte, der Weierstraß-Invarianten und der j-Lesart fuer kanonischen und positiv transportierten Eisensteinfall; A12 traegt also genau einen Eisenstein-Semantikzweig mit mehreren kontrollierten Repraesentanten statt mehrere roh nebeneinanderstehende Spezialfaelle
A12g	Folgeblock	`Modularity/WeierstrassInvariant`, `Modularity/JInvariant`, begleitend `Modularity/ModularBridge`	Orbit- und Proper-Aequivalenzinvarianz laufen nun ueber gemeinsame Provenienz, Orbitgleichheit bzw. deren kontrollierten Transport und nicht mehr ueber zusaetzliche Gleichheitsannahmen fuer g_2, g_3 oder Δ. Wo die repo-interne Tragfaehigkeit derzeit nur die Ankerorbits traegt, wird dies explizit in der Signatur eingeschraenkt; der Plan verlangt hier ehrliche partielle Saetze statt scheinallgemeiner Invarianzbehauptungen

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A12h	Folgeblock	`Shadow/EisensteinSeriesShadow`, `Shadow/JSeriesApprox`, begleitend `Notation`	der computable q-, Eisensteinreihen- und Approximationsstrang ist sprachlich und mathematisch entschaerft: Shadow-Objekte duerfen dieselbe Traegerlage oder denselben Repraesentanten teilen, aber keine semantische Herkunft simulieren. Zugleich ist der Schatten nun inhaltlich zu schaerfen: fuer die in A12c festgelegte Normierung sind die ersten q-Koeffizienten der Eisensteinreihen und daraus der erste echte computable j-Schatten proof-carrying abzuleiten, statt nur eine formale Hauptteil- oder Verpackungshuelse bereitzustellen. Mindestens ein kleiner Koeffizientenblock der j-Reihe – einschließlich Hauptterm und erster regulärer Koeffizienten – muss aus den gewaehlten Eisenstein-/Weierstraß-Daten berechnet werden, sodass `JSeriesApprox` nicht bloss Rhetorik, sondern operative Fourieroberflaeche ist. Satznamen und Exportbezeichnungen machen deshalb nur noch Alignment, Approximation oder gemeinsame Traegerlage sichtbar; insbesondere wird aus einem Schattenobjekt nicht rhetorisch ein semantisch gewonnenes Invariantenpaket
A12i	Folgeblock	`Modularity/WeierstrassInvariant`, `Modularity/JInvariant`, begleitend interne Hilfsmodule	weiterhin benoetigte Rohschichten sind vollstaendig internalisiert: rohe semantische Datenpakete, freie Weierstraß- oder j-Hilfskonstruktionen duerfen fuer lokale Beweisarbeit fortbestehen, erscheinen aber nur noch in `Internal`- oder `Provisional`-Namensraeumen und nicht mehr als empfohlene Consumer-API. Damit ist derived-only nicht bloss proseartig behauptet, sondern als Eigenschaft der sichtbaren Moduloberflaeche erzwungen
A12j	Folgeblock	`BuildAll`, `Notation`, flankierend Audit ueber die A12-public-Oberflaeche	der A12-Block ist erst jetzt in seiner derived-only Grundschicht geschlossen: Build und Exporte sind gruen, die oeffentliche API zeigt nur noch `SemanticCMProvenance`, daraus definierte Gewichte, die derived Weierstraß-Schicht und die daraus abgeleitete j-Lesart, waehrend Shadow und Internal sichtbar getrennt bleiben. Die weitere modulare Funktionsseite von j wird erst in A12k explizit nachgezogen; A13 und alle spaeteren arithmetischen Consumer duerfen semantische Invarianten ab hier nur noch ueber diese harte A12-public-Schnittstelle konsumieren. Ab A11b sind die hier genannten Modulpfade als Zielpfade der neuen Themenordner und nicht mehr als provisorische Flachstruktur zu lesen

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A12k	Folgeblock	`Modularity/ModularityLayer`, begleitend `Modularity/JInvariant`, `Modularity/WeierstrassInvariant`, `Modularity/ModularBridge`	die modulare Funktionsseite des A12-Blocks ist nun explizit gemacht: die aus der public-Provenienzschicht abgeleiteten Eisenstein-/Weierstraß-Daten tragen ihr Transformationsverhalten gegenueber der modularen Gruppenwirkung sichtbar mit, und die daraus konstruierte j-Lesart ist nicht nur ein Wert, sondern auf der aktuell getragenen Welt als invariante Modulfunktion bzw. als kontrolliert transportierbare modulare Groesse formalisiert. Minimal bindend ist hier nun mindestens ein eigener Invarianzsatz der Form „modular bzw. proper-aequivalent transportierte Daten liefern denselben j-Wert“; nach Moeglichkeit ist dies durch ein explizites Transformationsgesetz fuer die zugrunde liegenden Eisenstein-/Weierstraß-Bausteine zu unterfuettern und nicht nur durch einen nackten Endwertsatz. Wo die Repo-Lage dies erst fuer die Ankerfaelle oder die kleine Welt proof-carrying traegt, bleibt die Signatur ehrlich eingeschraenkt; die q-Serien-/Approximationsseite bleibt davon getrennte Fourier-/Schattenoberflaeche und wird nicht als Ersatz fuer den Modularitaetssatz gelesen
A13	Folgeblock	`Modularity/HilbertClassPolynomialSmall`, flankierend `Orders/ProperClassSet`, `CM/SemanticCMProvenance`, `Modularity/WeierstrassInvariant`, `Modularity/JInvariant`, begleitend `Modularity/ModularityLayer`	fuer die kleinen aktuell getragenen negativen Diskriminanten ist der erste Klassenpolynom-Block geschlossen: Hilbert-Klassenpolynome werden auf der semantischen Seite ausschliesslich ueber die in A12j/A12k festgezogene public- und Modularitaetsoberflaeche definiert, also ueber reduzierte Klassen, explizite `SemanticCMProvenance`-Objekte und die daraus abgeleitete Weierstraß/j-Schicht mitsamt ihrer eingeschraenkten modularen Transportseite. Ihr Grad stimmt mit der jeweils bereits geschlossenen kleinen Klassenzahl ueberein, und die Klassenzahl-Eins-Faelle sind von echten Mehrklassenfaellen sauber getrennt. Ein computabler Kleinwelt-, Serien- oder Numerikschatten darf getrennt danebenstehen, ersetzt aber weder semantische Provenienz noch die abstrakte Invariantenoberflaeche; insbesondere bleiben die klassischen elliptischen Punkte reine Ankerchecks der aktuell getragenen kleinen Welt und keine voreilige allgemeine Theorie beliebiger Diskriminanten. A13 steht bewusst noch vor der Rueckeinspeisung konkreter CNNA-Signaturen: hier wird zuerst die kleine semantische Ziel-API stabilisiert, die A14–A17 anschliessend aus dem operativen Spektralpfad heraus legal bedienen muessen
A14	Folgeblock	und die endliche Spektralwurzel `Core/SchurFactorExtraction`, begleitend `Core/QuadraticFactorCertificateStrong`	aus der bisherigen Witness-/Certificate-Schicht wird nun – bewusst erst nach Stabilisierung der kleinen Ziel-API in A12j/A12k und A13 – ein echter End-to-End-Extraktionspfad: quadratische Faktorkandidaten und zugehoerige Zertifikate werden aus der CNNA-Schur-/Charpoly-Seite selbst gewonnen, statt dauerhaft nur als manuell verpackte Seed-Daten aufzutreten. Mindestens der $L = 2$- und der aktuell erreichte Gauß-nahe Fall muessen unter Konsumption von A1b und A5b aus realen `ArithmeticSignature`-Daten in diesen Pfad einspeisen; semantische CM-, Gitter- oder Invariantenconsumer duerfen daran spaeter nur ueber explizite Bruecken anschliessen und nicht als verdeckter Ersatz fuer die operative Extraktionsseite auftreten. Erst ab hier ist die arithmetische Folgearbeit wieder sichtbar und kontrolliert an den operativen CNNA-Pfad rueckgebunden

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A15	Folgeblock	`Core/HigherLevelSchurFamily`, optional `Core/L4Signal`	die erste hoehere, bisher nicht sicher klassifizierte Faktor- und Diskriminantenfamilie jenseits des kleinen Gauß-/Eisenstein-Bereichs ist sauber isoliert. Insbesondere wird fuer den naechsten relevanten Level-Fall theorematisch festgehalten, ob ein imaginär-quadratischer, reell-quadratischer oder neuer Mischfall vorliegt, ohne vorschnell eine Klassenzahl- oder Ringidentifikation zu behaupten. Etwaige spaetere CM-, Gitter- oder j-seitige Leser bleiben auch hier strikt nachgelagert und dualstrang-diszipliniert; aus einer computablen Diagnose folgt noch keine semantische Invariantenidentifikation. Dieser Schritt dient als Drucktest gegen die bisherige quadratische Schliessung und markiert explizit, ob der existierende Order-Block ausreicht oder erweitert werden muss
A16	Folgeblock	`Orders/CNNAOrderBridge`	die erste echte End-to-End-Bruecke von konkreten CNNA-Signaturen zu quadratischen Ordnungsdaten steht: aus einer legalen `ArithmeticSignatureStrong` oder einem davon gewonnenen Faktorzertifikat folgt proof-carrying ein Objekt vom Typ `QuadraticOrderData`. Mindestens ein Eisenstein-artiger und ein Gauß-artiger Pfad muessen dabei als echte Beispiele des operativen Systems erscheinen; semantische CM- und Invariantenconsumer werden von hier aus nur ueber explizite Shadow/Spiegel-Bruecken gespeist. Dieser Schritt ersetzt jede lose rhetorische Rede von „versteckten Ringtypen“ durch eine explizite Transport- und Erkennungskette
A17	Folgeblock	`CM/CNNACMBridge`	die CNNA-Ordnungsbruecke wird bis zur modularen Orbitseite fortgesetzt: aus den in A16 gewonnenen quadratischen Ordnungsdaten folgen in der aktuell getragenen kleinen Welt einerseits computable Shadow-Objekte fuer den operativen Pfad und andererseits davon getrennte semantische CM-, Gitter- und j-seitige Invariantenobjekte nur ueber explizite Bruecken in die seit A12j verfestigte Provenienz-/Derived-API. Damit ist die modulare Seite nicht mehr nur komparative Perspektive, sondern formaler A-interner Consumer des CNNA-Arithmetikpfads; zugleich bleibt strikt getrennt, was bereits proof-carrying geschlossen ist und was nur numerischer oder heuristischer Schatten bleibt
A18	Folgeblock	`Examples/*`, flankierend `PillarA/BuildAll` und modulweise externe Dokumentation	der Arithmetikpfad ist ueber mehrere gruene End-to-End-Beispiele konkretisiert: Referenzfall, Variationsfall, erkannter Faktortyp, reduzierte Form, Ordnungsdaten sowie explizit getrennte Shadow- und semantische CM-/Gitter-/Invariantenstufen erscheinen als maschinengepruefte Beispiele im Repo; ein kleiner j- oder Klassenpolynom-Schatten bleibt klar als Schatten markiert. Externe Dokumentation darf diese Kette erklaeren, aber nicht ersetzen; die primaere Quelle bleibt der gruene Code
A19	Folgeblock	`MainTheorems`, `BuildAll`, `PillarA/BuildAll`, modulweise externe Doku	der A-interne Arithmetikpfad ist architektonisch geschlossen: die Hauptsaetze des Pfads sind in einer gerichteten Reihenfolge gebuendelt, `BuildAll` konsumiert die Arithmetic-Seite ohne Schatten-Topologie, und die Abgangsflaeche nach aussen ist klar als pre-handoff A-Folgepfad markiert. Zu diesem Zeitpunkt ist auditierbar, dass der Arithmetikblock weder Handoff ersetzt noch spaetere Pfeiler vorwegnimmt, sondern einen eigenstaendigen, proof-carrying A-Folgepfad von der endlichen Spektralwurzel ueber die operative Extraktions- und Ordnungsseite bis zu kleinen modularen Invarianten geschlossen hat, wobei die Shadow/Spiegel-Trennung ab A11/A11b bis in die gebuendelten Hauptsaetze explizit sichtbar und nicht stillschweigend eingeebnet bleibt

Phase	Block	Betroffene Module	Status / Muss am Phasenende wahr sein
A20	Folgeblock	`Horizons/DeterminantalZetaInterface`, optional `Horizons/GraphZetaShadow`, begleitend `Finite/DirichletLaplacian`, `DtN/DtNStabilized`	jenseits des aktuell aktiven A-Kerns ist die zetafunktionale Leserichtung als eigener Fernblock reserviert: sobald die operator- und determinantenartige Seite stabil genug getragen ist, darf eine kontrollierte Bruecke von endlichen bzw. regularisierten spektralen Objekten zu einer determinantielle oder Ihara-artigen Zeta-/L-Lesart eingezogen werden. Dieser Schritt ist ausdruecklich *nach* A19 nachgelagert, ersetzt keine der vorherigen arithmetischen Schliessungen und darf insbesondere nicht den Eindruck erzeugen, die aktuelle A-Welt enthalte bereits eine vollgeschlossene globale Zetafunktion des DtN-Pfads
A21a	Folgeblock	optional `Horizons/FiniteSimpleGroups/CyclicLayer`, flankierend externe Doku	der endliche-Gruppen-Horizont beginnt nicht pauschal, sondern mit der ersten von TomS bereits explizit ausgefuehrten unendlichen Klasse einfacher endlicher Gruppen: den zyklischen Gruppen ueber Primkoerpern. In der Planlogik ist dies noch kein aktiver Produktionspfad, sondern die erste sauber benannte Unterstufe des Spaethorizonts; sie stellt klar, dass ein eventueller spaeterer Gruppenanschluss zunaechst strukturell und nicht sofort ueber Darstellungen, Matrizen oder Moonshine zu lesen ist
A21b	Folgeblock	optional `Horizons/FiniteSimpleGroups/AlternatingLayer`, flankierend externe Doku und spaetere Pillar-Schnittstellen	als zweite Unterstufe folgt nun die Permutationsseite mit symmetrischen und alternierenden Gruppen. Fuer Hintergrund und Reihenfolge ist explizit auf TomS, Beitrag #53, `https://www.astronews.com/community/threads/sich-durch-die-mathematik-treiben-lassen.12586/post-154366` zu verweisen: dort wird die Erzeugung von Permutationen durch Transpositionen, die 3-Zykel-Erzeugung der alternierenden Gruppen sowie die Ausnahme $A(4)$ als nicht-einfache Sonderlage hervorgehoben. Planlogisch bedeutet dies, dass ein spaeterer simple-group-Anschluss vor jeder Moonshine-/Monster-Lesart zunaechst diese strukturelle Permutationsstufe sauber ausweisen muss
A21c	Folgeblock	optional `Horizons/FiniteSimpleGroups/LieTypeLayer`, flankierend externe Doku	erst danach ist die dritte von TomS benannte Grossklasse als eigener Horizont zu reservieren: Gruppen von Lie-Typ ueber endlichen Koerpern. Diese Schicht bleibt weiterhin klar nach A19 und nach A20 nachgelagert; sie ist kein Vorgriff auf spaetere B/C/D/E-Logik und begruendet insbesondere noch keinen bereits aktiven Darstellungs- oder Automorphiekanal im A-Kern
A21d	Folgeblock	optional `Horizons/FiniteSimpleGroups/SporadicHorizon`, optional `Horizons/MoonshineHorizon`, flankierend externe Doku und spaetere Pillar-Schnittstellen	die sporadischen Gruppen bilden die letzte, endliche Klasse des Simple-Group-Horizonts. Ein spezieller Moonshine-/Monster-Kanal darf hoechstens noch als spaeteste optionale Zusatzrichtung markiert werden und erst dann, wenn eine explizite Bruecke von der modularen/q-seitigen A-Welt ueber die vorangehenden finite-simple-group-Stufen tatsaechlich formal tragfaehig gemacht wurde. Gerade TomS #53 spricht noch nicht fuer einen direkten Sprung dorthin, sondern fuer eine gestufte Ausfaltung des Gruppenhorizonts

Bindende Nicht-Ziele des Arithmetikblocks. Weder A1b–A6b noch A7–A21d ziehen S10a oder den Scalar-Emergence-Block S15–S20 vor, ersetzen spaetere AQFT-/OQS-/Matter-Fachlogik oder duerfen durch einen vorschnellen Griff nach Primzahlen, Heegner-Zahlen, j-Numerik, zetafunktionaler Fernlesart oder Gruppenassoziationen semantisch ueberdehnt werden. Insbesondere rechtfertigt der von TomS #30 explizit benannte Pfad $\tau \mapsto \Lambda_\tau \mapsto g_2, g_3, \Delta \mapsto j$ gerade nicht, schon jetzt volle

analytische Modulfunktionstheorie, Klassenkoerpertheorie oder eine globale Heegner-Lesart in den aktiven A-Kern hineinzuziehen; er praezisiert zunaechst allein den modularen Teilblock A10–A12k und dessen Consumption in A13. Ebenso bedeutet TomS' Hinweis auf L-Funktionen und endliche einfache Gruppen *nicht*, dass Dedekind-/Primidealtheorie, determinantielle Zetaobjekte oder Moonshine nun als aktuelle Nahziele bereits bewiesen waeren; diese Richtungen sind mit A9b sowie A20 und A21a–A21d nur planlogisch reserviert und bleiben bis zu ihrer tatsaechlichen Formalisierung klar nachgelagerte Folgepfade. A1b–A6b haerten allein die belegbaren Luecken des aktuellen gruenden Anfangsstands; erst A16–A19 entscheiden, ob der quadratische Pfad die CNNA-Signaturen wirklich bis zur Ordnungs- und modularen Invariantenseite traegt. Bis dahin bleibt jede entsprechende globale Deutung Forschungslesart und nicht Hauptsatz.

16 Handoff-Architektur, Emergenzsemantik und Pillar-Isolation zwischen den Pillars

- **Backreaction wird nicht modelliert, sondern nur als Theorie-Folge zugelassen.** Ein Rueckwirkungspfad existiert genau dann, wenn ein tieferer Pfeiler einen Satz erst formulieren oder schliessen kann, nachdem ein hoeherer Pfeiler in der partiellen Ordnung emergiert ist. Jeder zulaessige Rueckwirkungspfad muss daher theorie-erzwungen sein; alles andere gilt als Architekturverstoss.
- **Pillars sind Emergenzstufen einer partiellen Ordnung.** Aussagen in einem tieferen Pfeiler – insbesondere in A – sind deshalb je nach Emergenzstatus verschieden zu lesen. Der Plan unterscheidet ausdruecklich prae-B/post-B, spaeter analog prae-C/post-C usw.; post-emergente Saetze duerfen nie so formuliert werden, als waeren sie schon prä-emergent verfuegbar gewesen.
- **IDEAL-ToC hat keine Backreaction.** Solange kein Subsystem gewaehlt ist, gibt es keinen Split, keinen hoeheren Pfeiler und damit keinen Rueckwirkungskanal. Die gerichtete Grundlesart lautet: Subsystemwahl ⇒ Split ⇒ Emergenz des naechsthoeheren Pfeilers ⇒ kanonischer Handoff ⇒ zulaessige Backreaction.
- **Kein globales Mega-Handoff pro Pillar.** Pro Pillar gibt es eine stabile `Export`-Flaeche als zielneutrale oeffentliche Oberflaeche; sie ist die Kupplung nach aussen und nicht die interne Arbeitsoberflaeche desselben Pfeilers.
- **Gerichtete Outputs und Inputs sind strikt getrennt.** In der nach S6c gueltigen Struktur stehen `Outputs/` fuer ausgehende Exportkanaele wie `A_to_B`, `A_to_C`, `A_to_D`, `A_to_E`; `Inputs/` stehen fuer die einzigen zulaessigen Rueckwirkungskanaele wie `B_to_A`, `C_to_A`, `D_to_A`, `E_to_A`.
- **Verbotene Rueckgriffe.** Kein Pfeiler darf seinen eigenen Output-Handoff intern als Ersatzpfad konsumieren, auf den Output eines hoeheren Pfeilers zugreifen, ueber Direktimporte in spaetere Pfeiler hineingreifen oder Rueckwirkung durch Umgehung der Input-Schicht abkuerzen.
- **Pillar A hat eine Sonderrolle.** A ist der einzige Pfeiler ohne fundamentaleren Input-Handoff. Ohne Subsystemwahl und Split bleibt A im reinen IDEAL-Modus ohne Rueckwirkung. Sobald spaetere Pfeiler emergieren, darf A Rueckwirkung nur ueber explizite Inputs wie `B_to_A` oder `C_to_A` erhalten.
- **Der strukturelle Cayley-Dickson-Pfad ist kein normales Handoff.** Beweispflicht I wird ab S6b als eigener A-interner Meta-Beweisstrang `PillarA/Structural/CayleyDickson/*` gefuehrt. Dieser Bereich ist Teil von Pillar A, aber weder globale Exportflaeche noch Rueckkanal. Innerhalb von `PillarA/Handoff` erscheint davon hoechstens eine duerre Abschluss- bzw. Ledger-Oberflaeche in `ProofObligationI/*`.
- **ScalarEmergence ist A-intern und statusgetrennt.** Der Bereich `PillarA/ScalarEmergence/*` ist ab jetzt ein A-interner Folgeblock fuer Prae-Numerik, Scalar-Schichtung und diagnostische bzw. theorematische Folgepfade. Er darf den A-Generator nicht als zweiten Root-Generator duplizieren

und keine ueberschiessende Schliessungsbehauptung transportieren; seine Rolle ist Statusdisziplin, Vergleichsoberflaeche, spaetere Exportdiagnostik und – wo sauber beweisbar – produktive Consumption des pre-handoff A-Kerns.

- **Beweispflicht II und III werden gerichtungsrein vorbereitet.** `ProofObligationII/*` ist in S6c strukturell angelegt und ab S10a inhaltlich oeffnungsfaehig. `ProofObligationIII/*` bleibt fuer den Prototyp einer theorie-erzwungenen Rueckwirkung von B/C nach A reserviert: die spaeter benoetigten Daten duerfen dann nur als `B_to_A` bzw. `C_to_A` auftreten.
- **Strukturregeln sind in Signaturen sichtbar zu machen.** Spaetere A-Saetze mit echter Rueckwirkung sind ueber explizite Inputdaten zu formulieren, etwa schematisch `theorem some_A_statement_postB (hBA : BToAInput ...) : ...`; direkte `import`-Abkuerzungen auf `PillarB` sind gerade nicht die zulaessige Form.
- **Für flankierende Diagnosepfade bleiben IDEAL- und REAL-Provenienz getrennt sichtbar.** Spaetere Pfeiler duerfen beide Schienen konsumieren und vergleichen, aber weder in A zurueckspeisen noch durch eigene A-Rekonstruktionen ersetzen.
- **Nachfolgende Pfeiler duerfen fehlende Werkzeuge eines frueheren Pfeilers nicht nachbauen.** Wenn B, C, D oder E A-seitige Diagnose-, Skalen-, Spektral-, Zustands-, thermische oder limesseitige Werkzeuge benoetigen, gehoeren diese in den Strengthening-Pfad von A und nicht in spaetere Consumer.

16.1 Aktueller Belegungsstand der gerichteten Handoff-Kanaele (nur jetzt besprochene Inhalte)

Die folgende Belegungsliste ist absichtlich strikt konservativ. Eingetragen wird nur, was im vorliegenden Schritt semantisch bereits festgelegt wurde. Alle uebrigen gerichteten Kanaele bleiben ausdruecklich strukturell reserviert, aber inhaltlich unbefuellt. Damit wird vermieden, dass spaetere Pfeilerbeziehungen schon jetzt durch Namenswahl oder Platzhaltersemantik vorweggenommen werden.

Semantische Vorsicht fuer Casimir-, Feynman- und Hurwitz-Lesarten. A-seitig werden keine Kernkanaele als `Casimir*`, `FeynmanDiagram*` oder als CNNA-interne `HurwitzCompletion*`-Schliessung geschlossen. Belastbar sind in A nur algebraische Interface-, Schur-/Resolvent-, Spektralgap-, Horizont- und Spektralshift-Seeds. Eine spaetere physikalische Identifikation als Casimir-Kraft, freie Energie, effektive Wechselwirkung oder diagrammatische Expansion bleibt Folgearbeit der spaeteren Pfeiler; ebenso bleibt der Hurwitz-Stop in A ein externer harter Randstein bzw. diagnostischer Seed und gerade keine bereits intern geschlossene CNNA-Identifikation des Endes der Komplementhierarchie. Falls ein duenner Wrapper benoetigt wird, darf `CasimirCandidateSeed` nur unter `Outputs/A_to_D/*` und nur explizit kandidathaft auftreten; `InterfaceMediatedInteractionSeed` ist dagegen als A-seitiger algebraischer Seed zulaessig. Die S10c-Module `DualObserverSeed`, `InterfaceEntropyBou` `SpectralGapSeed`, `SchurBlockSymmetrySeed`, `HorizonDefinition` und `CayleyDicksonSpectralShift` sind in diesem Sinn nur algebraische Vorstufen und keine bereits physikalisch identifizierten Endkanaele.

Output-Sicht (gerichtete Ausgaenge). Die folgende Liste beschreibt die gerichteten Exportkanaele aus Sicht des sendenden Pfeilers. Bei Kanaelen, deren eigentliche Fachschicht noch nicht ausgearbeitet wurde, wird hoechstens die strukturelle Reservierung festgehalten.

`A_to_B` befuellt mit `AlgebraicCoreSeed`, `SpectralStateSeed`, `DynamicsThermalSeed`, `BInterfaceSeed` und `DirectedLimitSeed`. Diese Schicht liefert B genau die A-seitig derived-only erzeugbaren algebraischen, spektralen, thermischen, dynamischen und schnittstellenbezogenen Vorstufen, jedoch noch keine B-interne modulare oder AQFT-Schliessung.

`A_to_C` befuellt mit `ChannelCouplingSeed`, `SysEnvSeed`, `RelativeEntropyFlowSeed`, `MismatchSeed`, `TailEliminationSeed`, `UpdateStepSeed` und `InterfaceMediatedInteractionSeed`. Diese Schicht liefert C die kanalische und interface-vermittelte Vorstufe, jedoch noch keine offene-System-, Lindblad-, Semigruppen- oder Nichtmarkov-Schliessung.

`A_to_D` befuellt mit `PreGeometrySeed`, `DirectedLimitGeometrySeed`, `ScaleClosureSeed`, `ParameterProvenanc` `CouplingSpectralShiftSeed`, `DualObserverSeed`, `InterfaceEntropyBound`, `SpectralGapSeed`, `SchurBlockSymmetrySeed`, `HorizonDefinition`, `CayleyDicksonSpectralShift` sowie dem ausdruecklich nur kandidathaften `CasimirCandidateSeed`. Diese Schicht liefert D die geometrie-, closure-, horizon- und regimebezogene A-Vorstufe, ohne bereits eine physikalische Casimir-Identifikation, eine Kraftinterpretation oder eine ausformulierte Projektionsregime-Physik zu schliessen.

`A_to_E` befuellt mit `ScalarEmergenceStatusSeed`, `QuaternionicDiagnosticSeed`, `OctonionicDiagnosticSeed` `HurwitzStopSeed` und `MatterPrerequisiteSeed`. Diese Schicht liefert E nur diagnostische, status- und voraussetzungsbezogene Vorstufen; Charge-, Gauge-, Matter- und Symmetrieschliessungen gehoeren weiterhin nicht nach A. Die S10c-Spektral- und Horizontseeds duerfen von E spaeter allenfalls sekundär konsumiert werden, sind im Plan aber primaer nicht als Matter-Handoff, sondern als D-regimebezogene Vorstufe deklariert.

`B_to_A` als gerichteter Rueckkanal strukturell reserviert; seine inhaltliche Zielbelegung wird in der Input-Sicht von A festgelegt. Eine weitere B-seitige Verpackungssemantik wird in diesem Schritt noch nicht vorweggenommen.

`B_to_C`, `B_to_D`, `B_to_E` strukturell reserviert, in diesem Schritt inhaltlich unbefuellt.

`C_to_A` als gerichteter Rueckkanal strukturell reserviert; seine inhaltliche Zielbelegung wird in der Input-Sicht von A festgelegt. Eine weitere C-seitige Verpackungssemantik wird in diesem Schritt noch nicht vorweggenommen.

`C_to_B`, `C_to_D`, `C_to_E` strukturell reserviert, in diesem Schritt inhaltlich unbefuellt.

`D_to_A` als gerichteter Rueckkanal strukturell reserviert; seine inhaltliche Zielbelegung wird in der Input-Sicht von A festgelegt. Eine weitere D-seitige Verpackungssemantik wird in diesem Schritt noch nicht vorweggenommen.

`D_to_B`, `D_to_C`, `D_to_E` strukturell reserviert, in diesem Schritt inhaltlich unbefuellt.

`E_to_A` als gerichteter Rueckkanal strukturell reserviert; seine inhaltliche Zielbelegung wird in der Input-Sicht von A festgelegt. Eine weitere E-seitige Verpackungssemantik wird in diesem Schritt noch nicht vorweggenommen.

`E_to_B`, `E_to_C`, `E_to_D` strukturell reserviert, in diesem Schritt inhaltlich unbefuellt.

Input-Sicht (gerichtete Eingaenge). Die folgende Liste beschreibt dieselben gerichteten Kanaele aus Sicht des empfangenden Pfeilers. Im aktuellen Stand sind nur die expliziten Rueckwirkungskanaele nach A inhaltlich belegt. Fuer B, C, D und E werden die A-seitigen Ausgaenge zwar semantisch gespiegelt, aber in diesem Schritt noch nicht als eigene Empfaenger-Fachlogik ausdifferenziert.

`B_to_A` befuellt mit `ComplementGNSInput`, `ModularConjugationInput`, `KMSInput` und `LocalNetClosureInput`. Diese Rueckkanaldaten sind genau die B-seitigen Zusatzdaten, ueber die spaeter post-B-Aussagen ueberhaupt erst theorie-erzwungen formulierbar werden duerfen.

`C_to_A` befuellt mit `ChannelBackreactionInput`, `EntropyProductionInput`, `TailEliminationInput`, `UpdateLawInput` und `NonMarkovInput`. Diese Rueckkanaldaten sind genau die kanalischen Zusatzdaten, ueber die spaeter post-C-Aussagen in A formuliert werden duerfen.

`D_to_A` befuellt mit `GeometryClosureInput`, `ParameterFixpointInput` und `RegimeRecoveryInput`. Diese Rueckkanaldaten sind genau die geometrie- und closureseitigen Zusatzdaten, ueber die

spaeter post-D-Aussagen in A formuliert werden duerfen.

E_to_A befuellt mit `ChargeSectorInput`, `MatterResponseInput` und `SymmetrySelectionInput`. Diese Rueckkanaldaten sind genau die matter- und symmetrieseitigen Zusatzdaten, ueber die spaeter post-E-Aussagen in A formuliert werden duerfen.

A_to_B, A_to_C, A_to_D, A_to_E auf der Empfaengerseite in diesem Schritt nur durch die obige Output-Sicht semantisch bestimmt; keine zusaetzliche B-, C-, D- oder E-interne Input-Umsortierung wird hier bereits festgelegt.

B_to_C, B_to_D, B_to_E, C_to_B, C_to_D, C_to_E, D_to_B, D_to_C, D_to_E, E_to_B, E_to_C, E_to_D strukturell reserviert, in diesem Schritt inhaltlich unbefuellt.

Arbeitsregel fuer die weitere Planfortschreibung. Neue Handoff-Namen duerfen kuenftig nur dann in diese Listen eingetragen werden, wenn entweder (i) ihre A-seitige derived-only Herkunft bereits geschlossen ist oder (ii) ihre spaetere Rueckwirkung als expliziter Inputkanal theorie-erzwungen benannt wurde. Rein heuristische physikalische Lesarten ohne geschlossene Herkunft oder ohne klaren Empfaengerkanal gehoeren nicht in die Handoff-Listen.

17 Zukünftige Arbeiten jenseits von Pillar A: Semantische Schärfung der Folgepfeiler

Die folgenden Punkte gehören *nicht* mehr in den abzuarbeitenden Strengthening-Hauptpfad von Pillar A. Sie sind als präzise Folgearbeiten jenseits von A einzuplanen, weil sie echte AQFT-, OQS-, Geometrie- oder Matter-Fachlogik tragen. Zugleich werden sie jetzt semantisch geschärft, damit der Übergang von A in spätere Pfeiler weder auf alte boundary-first Muster zurückfällt noch A-seitige Vorstufen später heimlich nachbaut. Maßgeblich ist dabei die Roadmap-Regel, dass A nur solche Samen liefert, die aus A-generierten Größen derived-only erzeugbar sind, während spätere Pfeiler diese Samen auf ihrer eigenen Fachschicht weiterverarbeiten. Vgl. die Roadmap zum A→B-Handoff, zum modularen B-Kern, zur Kanal- und Backreaction-Lesart in C sowie zu den späteren Seeds für D/E.

17.1 Zukünftige Arbeiten für Pillar B: AQFT aus A-Samen statt Bright-Recovery

- Die generische AQFT-Basis – `StarAlgebra`, `State`, `LocalNet`, `StateNet`, `GNS`, `KMS`, `RelativeEntropy`, `QuasiLocal/*` und die Haag–Kastler-nahe Grundschicht – bleibt eigenständige zukünftige Arbeit in Pillar B. Sie wird nicht nach A vorgezogen, muss aber die in A bereitgestellten algebraischen, thermischen, dynamischen und BInterface-Seeds direkt konsumieren.
- Das Legacy-Material aus `BoundaryMatrix*` bleibt für B ausdrücklich Extraktions-, Recovery- und Gate-Quelle, aber nie wieder Kernsemantik. Generische algebraische Lemmata, C*- und Zustandsaussagen dürfen extrahiert werden; ToC-gebundene Bright-Semantik ist in einen späteren Recovery-/Vergleichsstrang zu verschieben.
- Die B-seitige Konsumption der A-Exportfläche muss über explizite Brückenmodule laufen: insbesondere ein späteres `Handoffs/A_to_B/StarAlgebraBridge` und analoge Adapter für `LocalAlgebraNet`, `StateNet`, `ChannelNet` und quasilokale Vervollständigung. Diese Brücken liegen absichtlich außerhalb von A und B, damit weder A B importiert noch B A-interne Logik rekonstruiert.
- Der modulare B-Kern – `ComplementGNS`, `SeparatingProperty`, `TomitaTakesakiData`, `ModularConjugation`, `ComplementKMS`, `BisognanoWichmannSeed` – bleibt zukünftige B-Fachlogik. A liefert dafür nur die endliche Operatoralgebra, Gibbs-/Dynamikseeds, gerichtete Limesvorstufen und die sektorisierte Exportfläche. Falls daraus spaeter A-seitig neue Saetze erst schliessbar werden, duerfen diese gerade nicht ueber B-Imports, sondern nur ueber explizite Rueckkanäle `B_to_A` der in S6c vorbereiteten Input-Schicht zurueckwirken.

- Gerade fuer den Scalar-Emergence-Block ist B mehr als bloße Konsumschicht: falls die modulare Konjugation J in spaeteren B-Seeds antilinear und konjugationsartig als kanonische Operation auftritt, liefert dies einen zusaetzlichen externen bzw. B-seitigen Anker fuer die komplexe statt nur reell-positive Kopplung. Diese moegliche fuenfte Bedingung gehoert ausdruecklich nicht in A-S17, wohl aber als Folgearbeit an die Schnittstelle A→B.
- Besonders zu vermeiden ist jeder Rückfall in die alte Richtung "BoundaryMatrix zuerst, Komplementnetz spaeter". Die Roadmap fordert explizit `ComplementLocalNet` und `InterfaceLocalNet` vor jeder Recovery. B hat daher aus A-Sektorstruktur und BInterface-Seeds ein echtes Komplementnetz zu bauen, statt den Bright-Rand erneut zum impliziten Zentrum zu machen.

17.2 Zukünftige Arbeiten für Pillar C: Kanäle, Tail-Elimination und Backreaction

- Pillar C bleibt die Fachschicht für sektorielle Kanäle, Stinespring-, Lindblad-, Semigruppen- und Nichtmarkov-Lagen. A liefert dafür nur die algebraischen Seeds, `RelativeEntropyFlow`, `SectorChannels`, `SectorSysEnv` sowie die neuen Update-/Mismatch-/Tail-Elimination-Seeds.
- Die Roadmap verlangt ausdrücklich, Tail-Elimination als Kanal neu zu lesen und Backreaction als abgeleitete, nicht frei postulierte Zusatzdynamik zu behandeln. Genau deshalb werden `MismatchSeed`, `TailEliminationSeed` und `UpdateStep` in A vorbereitet, waehrend ihre offene-System-Interpretation, semigruppenartige Spezialisierung und Nichtmarkov-Verallgemeinerung zukünftige C-Arbeit bleiben.
- C darf `DerivedSpacetime` nicht als fertige Ontologie abschließen. Die Roadmap ordnet diese Richtung explizit als Vorbereitungsobjekt für D ein: aus Kanalinfluss wird emergente Kausal- und Geometriestruktur. C bleibt daher Kanal-, Entropie- und Rückwirkungspfeiler; die echte Geometrieschließung ist D vorbehalten. Gerade C ist damit zusammen mit B die vorgesehene Herkunftsschicht fuer die spaetere Beweispflicht III: wenn A spaeter zusaetzliche Sätze nur noch mit kanalisch/modular emergierten Daten formulieren kann, duerfen diese Daten ausschließlich als `C_to_A` bzw. in Kombination mit `B_to_A` erscheinen.
- Zu vermeiden ist jeder semantische Rückfall, bei dem der dunkle Sektor wieder nur als eliminierte Restumgebung erscheint. C hat die Interface-vermittelte Wechselwirkung kanalisch zu explizieren und aus A nur diejenigen Seeds zu konsumieren, die diesen Schritt vorbereiten.

17.3 Zukünftige Arbeiten für Pillar D und Pillar E: Closure, Geometrie, Symmetrie und Matter

- Die eigentliche physikalische Schließung der Parameter b, $L_{\max}$, β und sekundärer Regularisierungsterme liegt nach Roadmap nicht in A, sondern am D-Eingang. A macht deren Provenienz sichtbar; B/C liefern modulare und kanalische Daten; D schließt den Fixpunktkreis aus Backreaction, Zustandsdaten, Skalenwahl und effektiver IR-Dynamik. Diese Rueckkopplung ist aber nicht frei zu modellieren, sondern spaeter nur ueber die in S6c vorbereiteten Input-Handoffs und damit als theorie-erzwungener Pfad zuzulassen.
- Seeds wie `BisognanoWichmannSeed`, `NuclearitySeed`, `DHRSectorsSeed` und `FieldAlgebraReconstructionSe` bleiben ausdruecklich zukünftige B/D/E-Arbeit. Dasselbe gilt für `RelativeCauchyEvolutionSeed`, `HadamardStateSeed` und `LocalWickPolynomials`. Im Strengthening-Plan dürfen sie nur als Folgearbeit semantisch vorbereitet, aber nicht als scheinbar A-seitig schon geschlossen behauptet werden.
- Die im Scalar-Emergence-Block sichtbare Drei-Generationen-Lesart bleibt ebenfalls ausdruecklich Fernziel. Planlesbar ist nur die schwache Form: eine spaetere Korrespondenz Cayley-Dickson-Stufen ↔ Fermionengenerationen wuerde mindestens quaternionische und oktonionische Seeds sowie DHR-/Field-Algebra-Rekonstruktion voraussetzen; Literaturanker wie Günaydin/Gürsey 1973 und Furey 2015 sind dabei nur Orientierung, nicht bereits konsumierte Planbeweise.

- Der Hurwitz-Stop schliesst normierte Divisionsalgebren oberhalb von **O** als operative Fortsetzung aus; exzeptionelle Jordan-Algebren wie $J_3(\mathbf{O})$ umgehen diese Schranke, weil sie keine normierten Divisionsalgebren sind. Sie bleiben deshalb als moegliche, aber deutlich spaetere Matter-/Vereinheitlichungsfortsetzung offen und gehoeren nicht in S15–S20.
- Auch die Frage einer weitergehenden Substrat-Selbstbegruendung bleibt Folgearbeit statt A-Kernpflicht: durch `Foundation/SubstrateAnalysis` und `PillarA/VariationAnalysis` ist sie inzwischen jedoch als konkrete Vergleichs- und Rueckbindungsfrage formulierbar und nicht mehr bloß als vage Metafrage offen.
- Für E gilt dieselbe Semantikregel: A liefert Träger-, Algebra-, Zustands-, Dynamik-, Diagnose- und Limesvorstufen; Charge-, Gauge-, Matter-, CPT-, Kramers- und Spin-Statistik-Schichten entstehen erst auf stabiler modularer und kanalischer Unterlage in späteren Pfeilern.
- Der Legacy-Bestand darf auch hier nur als Extraktions- und Vergleichsquelle dienen. Jede spätere Portierung hat das Fachmodulprinzip des Refactors beizubehalten: pro Modul eine zentrale Aussage bzw. Fachlogik, keine kreuz und quer vermischten Derived-/Gate-/Example-Sammelbäume.

17.4 Semantische Leitregel für den Legacy-Einsatz

Der Legacy-Pfad von REALOQS ist für CNNA wertvoll, aber semantisch zu entzentrieren. Aus ihm dürfen allgemeine Mathematik, Gates, Bright-Recovery-Vergleichsobjekte und spätere Seed-Kandidaten systematisch extrahiert werden. Nicht zulässig ist dagegen, Legacy-Strukturen wegen ihrer faktischen Verfügbarkeit erneut zum impliziten Zentrum der neuen Stammtheorie zu machen. Dies betrifft besonders alte AQFT-Derived-Dateien, Bright-spezifische Dynamikableitungen und A→B-Zirkularitäten wie `AQFTSubstrate`. Die zukünftigen Folgepfeiler haben den Legacy-Bestand deshalb nur noch unter den Disziplinen *Extraktion*, *Recovery*, *Gatevergleich* und *Regression* zu nutzen.