Structural Semantics

Constraint Geometry, Scope Calculus,
and Resonant Models of Representation

Flyxion

A Research Monograph

Version 1.0

August 1, 2026

This work develops original mathematical constructions, formal definitions, and computational models. Where established results are discussed, appropriate references are provided.

"The aim of science is not things themselves, as the dogmatists in their simplicity imagine, but the relations among things."

— Henri Poincaré

Contents

Preface

The rapid development of modern machine learning has revived questions that have occupied philosophy, mathematics, linguistics, and cognitive science for more than a century. Systems capable of producing coherent mathematical arguments, computer programs, scientific summaries, and extended dialogue have blurred traditional distinctions between syntax, semantics, reasoning, and understanding. At the same time, the conceptual vocabulary used to discuss these systems has become increasingly heterogeneous: terms such as *understanding*, *representation*, *meaning*, *reasoning*, and *intelligence* are often employed without a common mathematical framework, leading to disagreements that arise as much from differences in definition as from differences in evidence.

The purpose of this monograph is to construct such a framework. Rather than beginning with phenomenological assumptions, behavioral benchmarks, or biological commitments, we begin with the mathematics of constrained transformation. The central question investigated throughout this work is whether semantic capacity can be characterized as a structural property of admissible transformations over bounded contexts.

The resulting framework, called Structural Semantics, combines ideas from topology, algebra, category theory, dynamical systems, signal processing, and machine learning, and engages process philosophy directly through a formal correspondence with Whitehead's ontology. The mathematical developments are separated carefully from their philosophical interpretations, allowing each to be evaluated on its own terms. Where a result depends on mathematics beyond what a given chapter needs to state it, the corresponding construction — from the λ-calculus and abstract rewriting theory through categorical, topological, sheaf-theoretic, and measure-theoretic foundations — is developed independently in the appendices rather than assumed. The work is intended equally for mathematicians, computer scientists, philosophers of mind, and researchers in mechanistic interpretability. No commitment to any particular metaphysical position is assumed.

Relation to earlier structuralism. This monograph does not replace the structural semantics developed within twentieth-century linguistics. Rather, it generalizes the structuralist methodology from systems of linguistic signs to systems of admissible computational transformations. The resulting framework shares with classical structuralism an emphasis on relational organization while introducing a different mathematical foundation based upon topology, algebra, category theory, and dynamical systems. The precise relationship between the two uses of the term is stated in remark 1.1 and developed at length in section 2.2.

Methodological Statement

Throughout this monograph, three distinct classes of claims are carefully distinguished, and every chapter closes with a status summary that sorts its content among them.

Mathematical claims consist of definitions, lemmas, theorems, and proofs concerning topology, scope algebra, category theory, and related formal structures; these are established by proof from stated hypotheses and by nothing else. *Computational claims* concern algorithms, diagnostic procedures, and models of transformer representations; these are empirical hypotheses whose validity depends upon experimental evaluation, and they are presented as such rather than as consequences of the mathematics. *Philosophical claims* concern interpretation: they discuss possible implications of the mathematical and computational developments but are not presented as logical consequences of the formal theory.

Maintaining this distinction is a central methodological principle of the present work. A theorem in Volume II never licenses a claim about consciousness or moral status; a diagnostic algorithm in Volume III is never treated as proven merely because it is well motivated; and a philosophical reading offered in Volume IV is never mistaken for a mathematical consequence of what precedes it.

How to Read This Monograph

Although the eighteen chapters are arranged sequentially, the book is organized into four relatively independent volumes, and readers are encouraged to enter at the volume suited to their interests rather than proceed straight through. Readers primarily interested in philosophical questions may begin with Volume I, which establishes the epistemic framing, historical lineage, and the boundary separating structural claims from phenomenological ones. Readers interested in the mathematical framework itself should concentrate on Volume II, where bounded scope geometries, the primitive algebra, the category of scopes, and the representation-invariance and process-geometry results are developed; the notation introduced in **??** is sufficient preparation, and the historical material is not a prerequisite. Readers interested in transformer analysis and computational diagnostics may read Volume III after becoming familiar with the basic scope formalism, since the diagnostics there are applications of, rather than alternatives to, the mathematics of Volume II. Volume IV situates the resulting framework within the broader landscape of philosophy, cognitive science, and applied mathematics, and states its limitations and non-claims explicitly. The appendices provide complete proofs, notation tables, implementation details, and supplementary derivations referenced throughout the text, and are consulted as needed rather than read in sequence.

Notation and Conventions

Bold uppercase symbols denote categories or structured spaces, calligraphic symbols denote operators, constraint systems, or families of sets, and lowercase Greek letters denote morphisms, transformations, or parameters. Once a symbol is assigned a formal meaning it is preserved throughout the monograph and is not reassigned in a later chapter; appendix H collects the complete registry.

Unless stated otherwise, $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$ denotes a bounded scope geometry, $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$ denotes the primitive event alphabet, and $\sigma : \mathrm{S}_i \to \mathrm{S}_j$ denotes an admissible scope morphism. These three lines anticipate Chapter 6 and are given here only so that forward references elsewhere in the front matter are legible; their full definitions, and the argument for why the alphabet has exactly four elements, are deferred to Part II.

The reconstruction of Whitehead's process ontology in Chapters 4 and 5 uses a second, disjoint set of symbols native to that literature: E for an actual occasion, D for antecedent data, P for a prehension, v for valuation polarity, S for a satisfaction state, and A for Subjective Form. These are kept visually distinct from the Scope Calculus notation deliberately: no identification between the two systems is assumed before Chapter 10, where the translation functor is constructed and any correspondence between them is established as a proposition rather than presupposed by shared notation.

Finally, a computational system is written $M = (\mathcal{D}, \mathcal{T})$, where $\mathcal{D}$ is a state space and $\mathcal{T}$ is the *set* of admissible transition operators acting on $\mathcal{D}$, each individual operator being written $\tau \in \mathcal{T}$ with $\tau : \mathcal{D} \to \mathcal{D}$. This corrects an earlier draft in which $\mathcal{T}$ was written as if it were itself a single function; the set reading is what the rest of the text relies on whenever it writes $\tau_i \in \mathcal{T}$.

Part I

Epistemic and Historical Boundaries

Chapter 1

Epistemic Framing, Disentanglement, and Primary Research Questions

Chapter Roadmap. This chapter establishes the epistemic foundations of the monograph. It defines the central object of study, separates structural semantic capacity from neighboring philosophical concepts, introduces the primary research questions, summarizes the principal contributions, and outlines the methodological strategy that governs all subsequent chapters.

The limits of my language mean the limits of my world.
— Ludwig Wittgenstein, *Tractatus Logico-Philosophicus*

1.1 Introduction

Few concepts have become simultaneously more important and more ambiguous than the notion of *understanding*. Within philosophy, cognitive science, linguistics, neuroscience, and artificial intelligence, the same word often refers to fundamentally different phenomena: subjective conscious awareness in one context, successful behavioral performance in another, internal representation, semantic competence, or predictive capability in others. Debates regarding machine intelligence frequently proceed without agreement concerning the object under discussion itself.

Recent advances in transformer-based language models have intensified this ambiguity. Systems capable of producing mathematically coherent proofs, generating software, translating between natural languages, summarizing scientific literature, and participating in extended technical dialogue have challenged long-standing intuitions concerning the relationship between computation and semantic understanding [8, 45, 55]. At the same time, philosophical objections remain influential, including the Chinese Room [50], the distinction between phenomenal and access consciousness [7], and various forms of embodied cognition questioning whether statistical computation alone can constitute genuine understanding.

The present work does not attempt to resolve the metaphysical question of consciousness. Instead it addresses a more precise question: is there a mathematically definable notion of semantic capacity that depends only upon the preservation of structural relationships during computation? If such a notion exists, it should admit formal definitions, rigorous proofs, computational implementations, and empirical evaluation independently of debates concerning subjective experience. We refer to this proposed framework throughout as *Structural Semantics*. The central hypothesis is that semantic capacity may be understood as the preservation of admissible structural relationships across transformations of bounded contexts; later chapters develop this proposal using topology, rewriting systems, category theory, dynamical systems, and signal processing.

Remark 1.1 (Terminological note). The phrase *structural semantics* has an established meaning in linguistics, referring to structuralist theories of lexical meaning developed by Saussure, Hjelmslev, Lyons, Greimas, Coseriu, and related authors. Throughout this monograph the term is used in a different, explicitly defined sense: a mathematical framework for the preservation of admissible structural transformations over bounded contexts. The present theory is inspired by the general structuralist tradition but is not a development of lexical field theory, componential analysis, or structural linguistics; section 2.2 situates the two usages against one another in detail.

1.2 Motivating the Problem

The history of theories of meaning exhibits a recurring pattern: philosophical accounts generally begin from one of three starting points. The first begins from conscious experience, regarding understanding as inseparable from phenomenal awareness or intentionality, as in phenomenology, process philosophy, and much contemporary discussion of consciousness. The second begins from observable behavior, as in functionalist approaches that characterize understanding in terms of successful interaction, prediction, or problem solving, treating internal implementation as secondary to externally measurable performance. The third begins from representation itself, asking whether internal structures preserve relationships sufficiently to support inference, abstraction, and generalization independently of their physical realization.

Each perspective captures an important aspect of cognition, but they address different questions, and difficulties arise when conclusions appropriate to one framework are transferred without modification into another: a behavioral criterion cannot by itself establish phenomenology, and the absence of phenomenological evidence does not immediately imply the absence of mathematically characterizable semantic structure. This observation motivates the methodological separation adopted here. Rather than collapsing the three viewpoints into a single definition, we introduce a formal framework that treats structural semantic capacity as an independent object of investigation, deliberately separated from the philosophical questions concerning consciousness that remain important but lie outside its scope.

1.3 Epistemic Separation

The first objective of the monograph is therefore methodological rather than mathematical. Before introducing scope geometry or category theory, it is necessary to distinguish several concepts that are frequently conflated in the literature: phenomenal consciousness, behavioral competence, veridical truth, intentional agency, and structural semantic

capacity. Only the last of these constitutes the primary object of mathematical investigation developed in subsequent chapters. The remaining concepts may interact with structural semantic capacity, but they are not identified with it; results proved within the Scope Calculus should not automatically be interpreted as results concerning consciousness, moral agency, or epistemic truth, which belong to different theoretical domains and require independent justification.

This separation serves two purposes. First, it permits the mathematical development to proceed independently of unresolved philosophical debates. Second, it makes empirical evaluation possible, since structural properties may be investigated directly through computational observation without requiring access to subjective experience.

1.4 Formal Foundations

Having established the methodological scope of the monograph, we now introduce the mathematical objects on which the remainder of the development is built, identifying only the minimal structures required to state the principal research questions precisely. Throughout the text a *system* is understood abstractly, with no assumption regarding its physical realization: it may be implemented biologically, digitally, symbolically, mechanically, or by any other computational substrate, and the definitions below are correspondingly representation-independent.

Definition 1.2 (Computational system)**.** A computational system is an ordered pair $M = (\mathcal{D}, \mathcal{T})$, where $\mathcal{D}$ is a non-empty state space and $\mathcal{T}$ is a set of admissible state-transition operators $\tau : \mathcal{D} \to \mathcal{D}$ acting on $\mathcal{D}$. No assumptions are made regarding continuity, discreteness, determinism, or stochasticity unless explicitly stated.

Remark 1.3. The abstraction deliberately encompasses finite-state automata, symbolic rewriting systems, neural networks, probabilistic graphical models, dynamical systems, and biological cognitive architectures. The mathematical framework developed throughout this monograph applies at this level of generality whenever the stated hypotheses are satisfied.

1.4.1 Representations

The notion of representation is equally general. Rather than committing to a particular implementation, we define a representation only through its structural role.

Definition 1.4 (Representation)**.** Let $M = (\mathcal{D}, \mathcal{T})$ be a computational system. A representation is an element $x \in \mathcal{D}$ together with the admissible transition structure induced by $\mathcal{T}$. Two representations need not possess identical internal encodings in order to support identical structural behavior.

This definition intentionally avoids identifying representations with vectors, symbols, neurons, graphs, or linguistic expressions; a representation refers only to the mathematical state on which the transition operators act.

1.4.2 Transformation

Understanding, regardless of philosophical commitment, concerns not isolated states but the evolution of states under transformation, so transformations rather than static representations become the primary object of investigation.

Definition 1.5 (Transformation sequence)**.** Let $M = (\mathcal{D}, \mathcal{T})$ be a computational system. A transformation sequence is a finite ordered family $T = (x_0, x_1, \dots, x_n)$ such that $x_{i+1} = \tau_i(x_i)$ for some $\tau_i \in \mathcal{T}$. The sequence T records the structural evolution of a representation through the state space.

1.4.3 Motivation for Scope Geometry

Transformation sequences alone are insufficient for characterizing semantic preservation, since two systems may undergo identical numbers of state transitions while preserving fundamentally different contextual relationships. A symbolic proof assistant may reject an invalid inference by enforcing logical typing constraints, whereas a neural language model may continue generating fluent text despite violating assumptions introduced earlier in the reasoning process; both systems perform transformations, yet only one explicitly preserves the contextual constraints governing the computation.

The central mathematical idea developed in this monograph is that such constraints may themselves be treated as geometric objects. Rather than asking whether a representation "contains meaning," we ask whether successive transformations preserve the admissible boundaries within which meaning is defined. This observation motivates *bounded scope geometries*, developed fully in Chapters 6 through 10; for the present chapter it suffices to observe that semantic preservation appears to depend not upon individual states but upon the structural integrity of the permissible transformations connecting them.

1.5 The Disentanglement Operator

The preceding discussion suggests that much of the disagreement surrounding semantic capacity originates not from conflicting mathematical results but from differences in the domains over which the term *understanding* is defined. Before constructing an algebra of semantic transformations, it is therefore necessary to specify the epistemic domain within which subsequent definitions operate. The purpose of the Disentanglement Operator is precisely this: rather than measuring semantic capacity, it separates semantic capacity from other properties that may coexist with it but are not logically implied by it, allowing later chapters to formulate structural results without simultaneously making claims concerning consciousness, truth, or agency.

Definition 1.6 (Disentanglement operator)**.** Let $\mathcal{P} = \{\Phi, \mathrm{T}, \mathrm{A}, \mathrm{F}, \mathrm{S}\}$ denote the collection of properties commonly associated with understanding, where Φ denotes phenomenal consciousness, T denotes veridical truth, A denotes intentional agency, F denotes syntactic fluency, and S denotes structural semantic capacity. The *Disentanglement Operator* $\mathfrak{D} : \mathcal{P} \to 2^{\mathcal{P}}$ identifies structural semantic capacity as an independent object of mathematical investigation by separating S from the remaining properties: throughout this monograph, $\mathrm{S} \not\equiv \Phi$, $\mathrm{S} \not\equiv \mathrm{T}$, $\mathrm{S} \not\equiv \mathrm{A}$, and $\mathrm{S} \not\equiv \mathrm{F}$.

Remark 1.7. Definition 1.6 establishes only logical independence; it does not imply statistical, causal, or empirical independence. Systems possessing high structural semantic capacity may also exhibit high behavioral competence, and biological systems may simultaneously possess structural semantic capacity and phenomenal experience. The present framework merely asserts that these properties are mathematically distinguishable and should not be identified without additional argument.

1.5.1 Interpretation

The distinction above serves as a recurring methodological principle throughout the monograph. Structural Semantics is not presented as a theory of consciousness: whether subjective awareness accompanies a particular computational architecture remains outside the scope of the present formalism and is revisited only in the philosophical discussion of Chapters 16 and 17. Nor is it a theory of objective truth: a system may preserve contextual relationships perfectly while reasoning within an entirely fictional, hypothetical, or mathematically inconsistent universe, since the framework evaluates the preservation of internal constraint structures rather than correspondence with external reality. Nor is it a theory of agency: goal-directed behavior, preference formation, intentional action, and moral status require additional theoretical machinery that is deliberately excluded from the present development. Finally, Structural Semantics is not equivalent to linguistic fluency: contemporary language models routinely generate syntactically well-formed sentences while simultaneously violating assumptions established only a few paragraphs earlier, a discrepancy that motivates the distinction between surface grammatical regularity and the preservation of admissible contextual structure.

Example 1.8 (Structural coherence without truth)**.** Consider a narrative set entirely within a fictional world governed by its own internal rules — for instance, that a particular journey between two named locations cannot be completed within a stated span of time. A system that continues to respect this constraint throughout an extended narrative, refusing continuations that would violate it, is exhibiting structural semantic capacity with respect to that narrative's admissibility conditions, regardless of the fact that neither the locations nor the constraint correspond to anything in physical reality. Conversely, a system that abruptly violates a constraint it had itself established earlier in the same narrative — introducing a character incompatible with rules already fixed for that fictional world — has produced a boundary violation in the sense of Chapter 6, independently of whether either version of the narrative is judged more "realistic." Structural Semantics evaluates the second kind of failure. It has nothing to say about the first.

1.6 Boundary Constraints and Bounded Contexts

The Disentanglement Operator isolates the domain of investigation but does not yet specify the mathematical structures through which semantic preservation is expressed. We therefore introduce the notion of a *boundary constraint*: informally, the conditions under which a state may enter, remain within, or leave a contextual region. Such constraints appear in type systems, formal grammars, topological boundaries, logical theories, and dynamical systems, and the present formulation deliberately abstracts over these examples, treating a boundary not as a physical surface but as a collection of admissibility conditions governing transformations of representations.

Definition 1.9 (Boundary constraint)**.** Let $M = (\mathcal{D}, \mathcal{T})$ be a computational system. A boundary constraint is a predicate $\mathcal{B} : \mathcal{D} \times \mathcal{T} \to \{0, 1\}$, where $\mathcal{B}(x, \tau) = 1$ indicates that the transition $x \xrightarrow{\tau} \tau(x)$ is admissible, and $\mathcal{B}(x, \tau) = 0$ indicates that it violates the structural constraints governing the current context.

Definition 2.7 intentionally avoids assigning any particular computational interpretation to $\mathcal{B}$: in different applications it may represent typing constraints, logical consistency conditions, physical conservation laws, permission systems, grammatical restric-

tions, or learned admissibility relations extracted from data. The subsequent mathematics depends only on the existence of such a predicate and not on its implementation.

1.6.1 Bounded contexts

Semantic reasoning rarely occurs within an unrestricted universe of states. Mathematical proofs introduce hypotheses, computer programs create lexical environments, conversations establish topics, and scientific models operate under explicitly stated assumptions; in each case reasoning proceeds within a bounded context whose admissibility conditions determine which transformations remain valid.

Definition 1.10 (Bounded context)**.** A bounded context is a pair $(\mathcal{D}, \mathcal{B})$ consisting of a state domain together with a boundary constraint predicate. Its admissible region is

$$\mathsf{Adm}(\mathcal{D}, \mathcal{B}) = \{x \in \mathcal{D} \mid \mathcal{B}(x, \tau) = 1 \text{ for every admissible } \tau\}.$$

The notation $\mathsf{Adm}(\mathcal{D}, \mathcal{B})$ recurs throughout the monograph, denoting informally the collection of states that satisfy every structural constraint currently imposed by the context.

1.6.2 Motivation for scope geometry

Bounded contexts remain insufficient for describing nested reasoning processes. Modern computational systems routinely manipulate multiple interacting contexts simultaneously: a theorem may invoke several nested lemmas, a compiler maintains independent lexical environments, transformer models preserve long-range dependencies across dozens of processing layers, and human reasoning shifts continuously between assumptions, sub-problems, hypothetical scenarios, and meta-level analyses. These examples suggest that semantic preservation cannot be understood solely through individual admissible regions, but requires a hierarchy of interacting contexts whose relationships are themselves subject to structural constraints. This motivates *scope geometry*: rather than viewing context as a single admissible set, it models reasoning as a collection of nested bounded regions connected by explicit admissibility-preserving transformations.

Definition 1.11 (Scope geometry, preliminary)**.** A scope geometry is a bounded context together with an explicitly specified collection of subordinate bounded contexts and the structural relations connecting them.

This definition serves only as a conceptual preview; the complete formal treatment, including the algebra of scope transformations, admissibility morphisms, and categorical structure, is developed systematically in Chapters 6 through 10. In particular, the full tuple $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$ is deliberately withheld until Chapter 6, so that it appears there as the culmination of the intervening development rather than as a symbol introduced before its mathematical necessity.

1.6.3 A preview of Structural Semantics

The progression established thus far is deliberately incremental: beginning with computational systems, we introduced representations, transformation sequences, boundary constraints, bounded contexts, and finally scope geometries, each concept extending

the previous one while remaining independent of any particular computational architecture. The central hypothesis explored throughout the remainder of this monograph may now be stated informally: semantic capacity is not identified with isolated representations, but is associated with the preservation of admissible structure across transformations acting on hierarchies of bounded contexts.

1.7 The Research Program

The preceding sections established the conceptual vocabulary on which the remainder of this monograph is built, but no substantial mathematical claims have yet been made; we have established only the domain within which such claims may be formulated. The objective of the remainder of this work is to determine whether semantic capacity can be characterized as a structural property of admissible transformations over bounded contexts, and this objective separates into four progressively more specialized research questions, concerning conceptual foundations, mathematical structure, computational implementation, and empirical verification respectively.

The first question concerns the object of study itself: can semantic capacity be defined independently of phenomenal consciousness, intentional agency, and behavioral performance? The purpose is not to deny the importance of consciousness but to ask whether semantic structure admits an independent mathematical description; an affirmative answer would imply that semantic capacity constitutes a legitimate object of mathematical investigation irrespective of one's preferred philosophical theory of mind. Volumes I and II develop the formal foundations required to address this question.

The second question concerns mathematical structure: which mathematical structures are necessary and sufficient to describe semantic preservation under transformation? Several candidate frameworks already exist, including topology, graph theory, rewriting systems, information geometry, category theory, and dynamical systems; rather than replacing these disciplines, the present work attempts to synthesize selected ideas from each into a unified theory of bounded semantic transformations, developed systematically in Chapters 6 through 10.

The third question concerns implementation: can structural semantic preservation be extracted from the internal representations of computational systems? Recent advances in mechanistic interpretability have demonstrated that large neural networks admit increasingly sophisticated internal analysis through activation trajectories, sparse feature models, residual stream decompositions, and circuit-level investigations [15, 42, 44], suggesting that semantic structure may itself become an observable computational quantity rather than merely a philosophical abstraction. Volume III develops one possible mathematical framework for such measurements.

Finally, every mathematical model intended to describe computational systems should generate consequences capable of empirical evaluation: can structural measures of semantic preservation generate falsifiable predictions distinguishing genuine contextual coherence from superficial statistical fluency? The monograph therefore concludes its technical development by constructing explicit diagnostic algorithms together with experimental protocols designed to evaluate the proposed framework, and these protocols are intentionally formulated so that negative results are as informative as positive ones; a framework that cannot in principle be falsified falls outside the scope of the present methodology.

1.8 Statement of Contributions

The mathematical developments presented throughout this monograph are organized around five principal contributions. The first is a formal Scope Calculus governing admissible transformations over bounded contexts, generated minimally by four primitive operations whose algebraic properties are developed in Chapters 7 and 9. The second is a structural definition of semantic capacity that depends only on admissibility-preserving transformations rather than on any particular computational substrate, permitting semantic equivalence to be investigated independently of implementation. The third is a categorical formulation of scope transformations together with a translation functor relating elements of Whiteheadian process philosophy to the Scope Calculus, developed in Chapter 10 as a comparative rather than reductive construction. The fourth is the formulation of the MEM|8 analytical framework for studying activation trajectories within transformer architectures, which treats transformed activation trajectories as objects of signal analysis using established tools from harmonic analysis and digital signal processing rather than proposing a physical wave ontology. The fifth is an empirical diagnostic program designed to evaluate the proposed mathematical structures, with explicit falsification criteria, statistical controls, computational complexity analyses, and implementation protocols developed in Chapters 14 and 15 and expanded in the appendices.

1.9 Methodological Roadmap

The remainder of this monograph is organized into four interconnected volumes. Volume I examines the historical and philosophical development of the concept of understanding, culminating in a precise characterization of the phenomenological boundary separating structural questions from questions concerning subjective experience. Volume II develops the mathematical foundations of Structural Semantics, beginning with bounded scope geometries and primitive operations and proceeding through category-theoretic constructions, normalization theorems, admissibility preservation, and representation invariance. Volume III shifts from abstract mathematics to computational application, reviewing established results from mechanistic interpretability before introducing activation trajectory analysis, the MEM|8 framework, and empirical diagnostic procedures for contemporary neural architectures. Volume IV situates the resulting framework within the broader landscape of philosophy, cognitive science, artificial intelligence, and applied mathematics, concluding with explicit limitations, non-claims, and open research questions. The appendices provide complete proofs, notation tables, implementation details, and supplementary mathematical derivations referenced throughout the main text.

1.10 Why Structure Rather Than Meaning?

Before turning to the historical survey, it is worth pausing on a choice that has been implicit throughout this chapter: the theory developed here does not begin by asking what meaning *is*. It begins by asking what relations among representations remain invariant as those representations are transformed. This is a substantive methodological decision, and it is worth making the reasoning behind it explicit rather than leaving it to be inferred from the sequence of definitions.

A direct definition of meaning inherits every disagreement already present in the literature surveyed in Chapter 2: whether meaning requires reference, mental content, use, verification conditions, or truth conditions is itself a live dispute, and any definition chosen at the outset would import that dispute into the formal development rather than resolving it. Structure, by contrast, is comparatively tractable. Whether a transformation preserves a given boundary, whether two admissible sequences reduce to the same normal form, whether a projection preserves admissibility — these are questions with determinate answers once the relevant objects are specified, regardless of one's antecedent theory of meaning.

This is not a claim that structure exhausts meaning, or that a full theory of meaning could be built from structural invariants alone. Chapter 17 states explicitly that Structural Semantics does not attempt such a reduction. The claim is narrower and more defensible: whatever else meaning involves, semantic systems exhibit organization that persists across representational change, and that organization can be studied on its own terms, prior to and independently of settling what meaning ultimately is. Beginning with structure therefore is not a wager that structure is all there is. It is a decision about where a tractable mathematical investigation can honestly start.

1.11 Concluding Remarks

The principal objective of this introductory chapter has been methodological rather than technical. Before constructing a formal theory of Structural Semantics, it has been necessary to delimit the domain within which the theory is intended to operate. Beginning with the observation that the term *understanding* possesses multiple incompatible meanings across philosophy, cognitive science, and artificial intelligence, we argued that many contemporary disagreements arise from treating these distinct meanings as though they referred to a single mathematical object. Rather than attempting to resolve longstanding debates concerning phenomenal consciousness or intentionality, the present work isolates structural semantic capacity as an independent object of investigation.

The resulting framework proceeds from computational systems to representations, from representations to transformations, from transformations to boundary constraints, and finally toward bounded scope geometries; each concept introduces precisely the mathematical structure required for the next while avoiding unnecessary assumptions regarding implementation or physical substrate. Consequently, no substantive claims concerning transformer architectures, neural representations, or signal-processing models have yet been made, since those developments depend on mathematical results not yet established. Maintaining this logical order is an important methodological principle throughout the monograph.

Status of Results: Chapter 1

Established background. This chapter reviewed established literature concerning philosophy of mind, artificial intelligence, cognitive science, and machine learning, including classical discussions of understanding [7, 50, 54], predictive processing [11, 18], modern deep learning [22, 55], and mechanistic interpretability [15, 44].

Formal developments. The chapter introduced the Disentanglement Operator, Boundary Constraint, Bounded Context, and preliminary Scope Geometry as original defini-

tions establishing the mathematical domain of Structural Semantics, and stated the primary research questions and overall methodological architecture of the monograph.

Empirical hypotheses. None tested here. The only forward-looking claim is that structural semantic preservation should admit computational measurement independently of behavioral fluency, a proposal developed formally in Volume III and subjected to explicit falsification protocols in Chapter 15.

Chapter 2

Historical Foundations

Chapter Roadmap. This chapter reconstructs the intellectual lineage from which Structural Semantics emerges, from Saussure through mechanistic interpretability, identifying for each tradition what it contributes and what remains to be formalized.

The value of a sign results solely from the simultaneous presence of other signs.
— Ferdinand de Saussure, *Cours de Linguistique Générale*

2.1 Introduction

The purpose of this chapter is analytical reconstruction rather than mere historical exposition. No existing framework discussed here is dismissed as incorrect; rather, each contributes an important structural component later incorporated into the Scope Calculus of Volume II. Structural linguistics contributes relational organization, formal semantics contributes compositional interpretation, process philosophy contributes dynamic evolution, category theory contributes compositional abstraction, and mechanistic interpretability contributes empirical access to internal representations. The central thesis of this chapter is that these traditions become considerably more compatible once semantic organization is treated as a problem of admissible structural transformation rather than static representation. Throughout, we ask of each tradition which mathematical object is taken to be primitive, how semantic organization is represented, and which structural properties remain invariant under transformation.

2.2 The Structuralist Tradition

The expression *structural semantics* has a well-established history within twentieth-century linguistics. Beginning with Ferdinand de Saussure, structuralist approaches emphasized that the meaning of linguistic units derives not from isolated reference but from their position within systems of relational differences [13], motivating a body of work including lexical field theory, componential analysis, relational semantics, and later structural linguistics [12, 23, 26, 36]. Despite their differences, these approaches share the methodological commitment that semantic phenomena are explained by analyzing the

relational organization of linguistic structures rather than treating individual words as independent semantic atoms.

The present monograph adopts this structuralist insight while extending it beyond linguistic objects: rather than studying lexical systems, semantic fields, or componential features, it investigates admissible transformations over bounded computational contexts, so that the primary mathematical objects become transformations, boundary constraints, and scope geometries rather than lexical relations. The adjective "structural" therefore retains its original emphasis on relations, but the mathematical domain is fundamentally different: where classical structural semantics studies relations among signs, Structural Semantics as developed here studies relations among admissible computational transformations.

Classical Structural Semantics	Structural Semantics (this work)
Lexical units	Computational states
Semantic fields	Bounded scope geometries
Sense relations	Admissible transformations
Componential features	Boundary constraints
Lexical structure	Constraint geometry
Meaning relations	Structural preservation

Table 2.1: Comparison between classical structural semantics and the framework developed in this monograph.

Definition 2.1 (Classical structural semantics)**.** Classical structural semantics denotes the family of linguistic theories that analyze meaning through systems of structural relations among linguistic units, including lexical fields, semantic features, and relational networks [13, 23, 26, 36].

Definition 2.2 (Structural Semantics, this work)**.** Throughout this monograph, *Structural Semantics* denotes the study of semantic preservation under admissible transformations of bounded computational contexts. Unless stated otherwise, the term refers to this mathematical framework rather than to definition 2.1.

2.3 Ferdinand de Saussure and Relational Structure

Modern structural approaches to language originate largely with Saussure's posthumously published *Cours de Linguistique Générale* (1916) [13]. Saussure rejected the view that linguistic meaning is intrinsic to isolated words, arguing instead that linguistic values arise from systems of mutual distinction: a sign consists of a signifier and a signified, neither possessing independent meaning in isolation, with meaning emerging through the network of differences separating one sign from every other sign in the system. From the perspective developed in this monograph, Saussure's insight may be read as an early recognition that semantic systems are constraint systems, in which meaning is preserved not because individual symbols remain unchanged but because admissible structural relationships remain stable under transformation.

Observation 2.3 (Relational priority). Saussure's structural linguistics assigns explanatory priority to relations rather than isolated linguistic entities. The Scope Calculus extends this principle by treating admissible transformations of those relations as the primary mathematical objects of investigation.

Saussure's framework nevertheless remained largely descriptive: it introduced no explicit algebra governing semantic transformations, no theory of nested contextual organization, and no mathematical account of how structural relationships evolve during reasoning or computation. These omissions lay outside the objectives of early structural linguistics, but they identify precisely the mathematical questions addressed later in this monograph, beginning with the formal development of bounded scope geometries in Chapter 6.

2.4 Louis Hjelmslev and Formal Structural Systems

Louis Hjelmslev sought to give structural linguistics a more rigorous formal foundation, attempting in his *Prolegomena to a Theory of Language* (1943) to construct linguistics as an autonomous discipline based on explicit structural principles rather than psychological or historical explanation [25]. Central to his account is the distinction between the planes of expression and content, each possessing its own internal organization, with linguistic meaning arising through their systematic correlation; and the distinction between form, the abstract organization governing a system, and substance, the material realization through which that form is implemented, so that the same structural organization may be realized through different physical media without altering the underlying linguistic system. This distinction foreshadows the Representation Invariance Theorem of Chapter 9, which shows in a different mathematical setting that semantic equivalence depends on preserved structural transformations rather than on a particular representational substrate.

Hjelmslev's theory remains primarily descriptive rather than operational: it introduces no explicit notion of admissible state transformation, no rewriting calculus governing structural evolution, and no topology of contextual boundaries. The Scope Calculus of Volume II may be understood as extending structural linguistics from static organization toward a fully operational theory of constrained semantic dynamics.

2.5 John Lyons and Structural Semantics

The expression *structural semantics* entered modern linguistic usage primarily through the work of John Lyons [35, 36], who shifted attention from the internal organization of linguistic signs toward the systematic organization of lexical meaning itself, arguing that lexical meaning is determined by the network of semantic relations within which each lexical item participates rather than by isolated dictionary definitions. Lexical field theory studies collections of semantically related expressions whose meanings are mutually constrained, and Lyons further emphasized systematic sense relations such as synonymy, antonymy, hyponymy, incompatibility, entailment, and meronymy, which organize vocabulary into structured semantic networks and may be read as constraints restricting the admissible configurations of lexical organization.

The present monograph agrees with Lyons that semantic organization is relational, but extends this observation in two directions: the primitive objects of the theory are bounded contextual scopes rather than lexical items, and the primary concern is not merely the existence of semantic relations but the admissible transformations that preserve those relations during computation. Whereas structural semantics in the linguistic tradition describes the organization of semantic systems, the Scope Calculus seeks to describe the dynamics of semantic organization itself.

2.6 The Shift from Static Structures to Dynamic Structures

The three traditions surveyed so far — Saussure, Hjelmslev, and Lyons — share a family resemblance worth naming explicitly before the chapter turns to Frege, Peirce, and Whitehead. Each studies a *fixed* relational system: a language at a given moment, whose signs stand in relations of difference, whose planes of expression and content are internally organized, whose lexical items occupy a stable network of sense relations. The object of analysis is the system's synchronic organization, not its evolution. Saussure himself insisted on this separation, distinguishing the synchronic study of a language state from the diachronic study of its historical change, and largely restricting structural analysis to the former.

This emphasis was appropriate to the questions classical structuralism asked. It becomes a limitation only relative to a different question, the one this monograph asks: not how a system of distinctions is organized at a moment, but how that organization behaves as the system undergoes transformation. A bounded scope geometry, introduced formally in Chapter 6, is in one respect a direct descendant of the Saussurean relational system — it too specifies a structure of admissible distinctions rather than treating entities as meaningful in isolation. What it adds is that the geometry is defined together with the transformations that act on it, so that admissibility becomes a property of transitions rather than a property of a single static configuration.

The remaining sections of this chapter — Frege's compositional combination of meanings, Peirce's semiosis as a chain of successive interpretants, and above all Whitehead's replacement of substance with process — can be read as successive steps away from purely synchronic description and toward treating transformation itself as the primary object of study. None of them frames the shift in exactly Saussure's terms, and none anticipates the Scope Calculus specifically. But by the time the chapter reaches Whitehead, the historical trajectory it traces has already moved a considerable distance from the static relational systems with which it began, and Part II inherits that trajectory rather than starting it.

2.7 Gottlob Frege and Compositional Semantics

Although Frege predates twentieth-century structural linguistics, his work established many of the foundations of modern formal semantics [17]. His distinction between sense (*Sinn*) and reference (*Bedeutung*) observes that two expressions, such as "Morning Star" and "Evening Star," may refer to the same object while presenting it under different conceptual descriptions, so that semantic analysis cannot be reduced solely to reference. Frege's Principle of Compositionality further holds that the meaning of a complex expression depends on the meanings of its constituents and the rule by which they are combined, a principle underlying formal semantics, type theory, programming-language semantics, and modern logic.

The Scope Calculus accepts that semantic organization is compositional, but generalizes compositionality beyond syntactic construction: rather than asking only how meanings combine, it investigates whether the transformations acting on those meanings preserve admissible structural constraints, so that composition becomes one instance of a broader mathematical problem concerning the preservation of semantic organization under transformation. Frege's framework primarily concerns the interpretation of completed expressions; it provides no explicit theory of evolving contextual states and no algebra governing semantic transformations, gaps that the Scope Calculus is built to

address while building directly on the compositional insight.

2.8 Charles Sanders Peirce and Triadic Semiotics

Charles Sanders Peirce developed a broader semiotic framework intended to explain how signs function across reasoning, communication, scientific inquiry, and cognition [46, 51], rejecting the view that signs exist merely as binary relations between words and objects. Every act of signification instead involves a triadic relation of sign, object, and interpretant, and because an interpretant may itself function as a new sign, interpretation may propagate through a chain Peirce called *unlimited semiosis*: meaning develops through an evolving network of successive interpretations rather than a single isolated semantic event.

Peirce's semiotics anticipates several themes developed later in this monograph, particularly its emphasis on process over static representation, but his interpretant remains a philosophical concept describing semantic mediation rather than an explicit mathematical operator. The Scope Calculus instead describes the structural constraints under which interpretation may evolve without producing contextual inconsistency, so that interpretation becomes one possible realization of a more general theory of admissible semantic evolution. Peirce's emphasis on process provides an important bridge to Whitehead, examined next, whose metaphysics treats process itself as the primitive constituent of reality.

2.9 Alfred North Whitehead and Process Philosophy

Whitehead's *Process and Reality* (1929) sought to replace traditional substance metaphysics with a comprehensive process ontology [56]. Where classical ontologies treat objects as persisting substances undergoing accidental modification, Whitehead argued that the fundamental constituents of reality are events of becoming, with stability emerging from recurring patterns of organized process rather than from immutable substance — an inversion from objects to processes that becomes central to the categorical translation developed in Chapter 10.

The primitive unit of Whitehead's ontology is the *actual entity* or *actual occasion*: a transient event that arises, integrates information from prior occasions through *prehension*, achieves completion (*satisfaction*), and contributes to future processes. Whitehead distinguishes positive prehensions, which contribute information to the developing entity, from negative prehensions, which exclude it, and calls the process by which multiple prehensions integrate into a single completed entity *concrescence*. Among these components, *subjective form* — the manner in which a datum is experienced during prehension — is explicitly phenomenal, and cannot simply be removed without altering the ontology; chapters 4 and 5 ask whether the structural organization of prehensions can be formalized independently of subjective form, and answer affirmatively.

Whitehead's framework shares with the present work the conviction that transformation is more fundamental than static state and that complex structures emerge from compositions of simpler relations, but it develops a metaphysical ontology intended to describe the general structure of reality, whereas the Scope Calculus develops a mathematical theory of admissible transformations acting on bounded contextual structures. The present work therefore neither adopts nor rejects Whitehead's metaphysical conclusions, extracting instead those structural components capable of rigorous mathematical

formalization, a project undertaken in full in Chapters 4, 5, and 10.

2.10 John Searle and the Chinese Room

Among the most influential critiques of computational theories of mind is Searle's Chinese Room argument [50], directed against the claim that the correct execution of a computer program is itself sufficient for understanding. Searle imagines a person who does not understand Chinese, enclosed in a room with a rule book, manipulating Chinese symbols according to purely formal instructions and producing output indistinguishable from that of a fluent speaker; to external observers the room appears to understand Chinese, but Searle argues that no genuine understanding exists, since the individual manipulates symbols syntactically without attaching semantic significance to them. Syntax alone, on this view, is insufficient for semantics.

The present monograph neither accepts nor rejects Searle's conclusions regarding phenomenal consciousness, arguing instead that the Chinese Room addresses a different question from the one investigated here: Searle asks whether formal symbol manipulation is sufficient for subjective understanding, while the Scope Calculus asks whether semantic organization can be mathematically characterized through the preservation of admissible structural transformations. These questions are logically independent, and this distinction motivates the epistemic separation introduced in Chapter 1 and developed formally in Chapter 5. Rather than answering Searle's question directly, Structural Semantics replaces the binary question of whether a machine understands with the more mathematically tractable question of which structural properties of a computational system remain invariant under admissible semantic transformation.

2.11 Ned Block and the Separation of Consciousness

Ned Block's distinction between *phenomenal consciousness* and *access consciousness* [7] clarified that many philosophical disagreements concerning artificial intelligence arise because these fundamentally different notions of consciousness are routinely conflated. Phenomenal consciousness refers to subjective qualitative experience, what Nagel called "what it is like" to be a conscious organism [41][1]; access consciousness, by contrast, concerns the availability of information for reasoning, planning, memory, verbal report, and behavioral control, and is defined functionally, admitting computational description far more readily than phenomenal consciousness.

Structural Semantics intersects Block's framework at a carefully delimited point: the Scope Calculus is not a theory of phenomenal consciousness, nor a complete theory of access consciousness, but investigates a third mathematical object, the geometry of admissible semantic transformations. Information accessibility may facilitate structural semantic organization, but structural semantic organization is defined independently, through the preservation of admissible boundaries (definition 1.6), so that the present framework neither requires nor denies phenomenal consciousness and does not require any particular cognitive architecture for access consciousness.

[1]The expression itself predates Nagel's 1974 essay: it appears in B. A. Farrell's 1950 paper *Experience*, and earlier still in Wittgenstein's remarks of 1946–47 and Russell's 1926 *Encyclopaedia Britannica* entry on psychology. Nagel's essay is credited here, as it standardly is, for making the phrase central to the philosophy of mind and the explanatory gap, not for originating it.

2.12 Predictive Processing and the Free Energy Principle

During the early twenty-first century, cognitive science increasingly modeled intelligence as a continuous process of prediction, error correction, and dynamical adaptation. Predictive Processing, developed by Clark, Hohwy, and others [11, 28], and the closely related Free Energy Principle proposed by Friston [18, 19], both treat cognition as active inference whose objective is the maintenance of coherent internal models under uncertain sensory information, describing cognition through probabilistic inference, dynamical systems, and Bayesian optimization rather than explicit logical manipulation. Perception is understood not as passive reception but as continual generation of predictions compared against observations, with cognition inferring the posterior $P(x \mid y)$ over latent causes x given observations y. Friston generalizes this into the minimization of an upper bound on Bayesian surprise called variational free energy,

$$F(q) = D_{\mathrm{KL}}(q(x) \,\|\, P(x \mid y)) - \log P(y),$$

where $q(x)$ is an internal recognition density approximating the true posterior; minimizing F approximately minimizes surprise while improving the accuracy of the internal generative model, unifying perception, learning, action, and homeostatic regulation. Active Inference extends the framework so that an organism may also act on its environment to reduce prediction error, replacing stimulus-response models with a closed feedback loop between organism and environment.

Predictive Processing unifies perception, learning, memory, and motor control within a common inferential framework, accommodates uncertainty naturally, scales to hierarchical architectures, and admits quantitative modeling; but although it explains how internal models evolve under prediction error, it provides comparatively little formal machinery describing the organization of semantic structure itself, since prediction error measures numerical discrepancy rather than the admissibility of contextual transformations or the preservation of semantic boundaries. Structural Semantics shares with Predictive Processing an emphasis on internal organization and stability under continual transformation, but its primary mathematical object differs: Predictive Processing minimizes variational free energy over probability distributions, while Structural Semantics studies admissible transformations over bounded scope geometries, with prediction error becoming only one possible signal influencing boundary operations rather than the defining quantity itself. The two approaches are therefore potentially complementary, a relationship revisited systematically in Chapter 16.

2.13 Mechanistic Interpretability

The rapid success of transformer architectures introduced a new scientific challenge: while large language models demonstrated remarkable behavioral competence, the internal mechanisms producing this behavior remained largely opaque. Mechanistic interpretability emerged as a research program seeking causal explanations grounded directly in the internal organization of the model, rather than treating neural networks as inscrutable black boxes evaluated only on input-output performance.

Modern transformer architectures consist of repeated blocks of multi-head attention, feed-forward networks, normalization, and residual connections [55]; if the residual stream at layer l is $\mathrm{x}_l \in \mathbb{R}^{d_{\text{model}}}$, the forward computation is the iterative transformation $\mathrm{x}_{l+1} = T_l(\mathrm{x}_l)$, forming a trajectory through a high-dimensional representation space rather than a sequence of explicitly symbolic operations. Recent work by Olah, Elhage,

Nanda, and collaborators suggests that many such computations can be understood in terms of interacting computational circuits — induction heads, copy circuits, name-mover circuits, arithmetic circuits, factual retrieval pathways, and long-range attention mechanisms among them [15, 42, 44] — indicating that distributed neural computation often possesses substantial internal organization despite the absence of explicit symbolic programming. Techniques including activation patching, causal tracing, feature visualization, sparse autoencoders, attention intervention, linear probing, and circuit analysis [15, 38, 44] attempt to identify stable computational features whose causal manipulation produces predictable behavioral changes, with the overall objective explanatory rather than merely descriptive.

Despite substantial progress, current interpretability techniques remain fragmented: many methods identify individual features, neurons, or circuits without a unified mathematical description of contextual organization across an entire computational trajectory, and no generally accepted algebra currently exists for describing admissible semantic transformations across changing internal representations. Structural Semantics builds on the objectives of mechanistic interpretability while proposing a different abstraction: individual neurons, attention heads, sparse features, or circuits are not themselves treated as semantic units, but constitute one possible substrate whose state transitions may be projected onto bounded scope geometries, so that the primary mathematical object shifts from individual network components to admissible transformations preserving contextual organization. This motivates the Representation Invariance Theorem of Chapter 9, which establishes conditions under which different computational substrates may be considered structurally equivalent within the Scope Calculus.

2.14 Historical Synthesis

The historical survey reveals three broad explanatory traditions. The first emphasizes subjective experience as the defining feature of understanding, including phenomenology, portions of process philosophy, and accounts centered on consciousness or intentionality. The second emphasizes functional organization, identifying understanding through behavioral competence or successful interaction with the environment, as in cybernetics, functionalism, and contemporary AI benchmarks. The third emphasizes structural organization, investigating the internal organization of representations and the transformations that preserve them; although elements of this viewpoint appear throughout structural linguistics, information theory, category theory, and mechanistic interpretability, no single unified mathematical framework currently develops this approach systematically, and the present monograph is situated within this third tradition.

Existing theories successfully explain many individual aspects of semantic organization — structural linguistics describes relational organization within languages, category theory studies structure- preserving mappings, predictive processing models adaptive inference, and mechanistic interpretability analyzes internal computation — but none develops a unified algebra describing how semantic contexts themselves transform while preserving admissible organizational constraints. This gap motivates the formal developments beginning in Part II. Structural Semantics does not seek to replace the traditions surveyed in this chapter, but to synthesize insights from several of them: the primacy of relational organization from structural linguistics, the study of structure-preserving transformations from category theory, the emphasis on dynamic evolution from process philosophy, the importance of continual adaptation from predictive processing, and the commitment to analyzing internal computational organization from

mechanistic interpretability.

Status of Results: Chapter 2

Established background. Historical developments in structural linguistics, process philosophy, philosophy of mind, predictive processing, active inference, and mechanistic interpretability [7, 11, 13, 15, 18, 36, 42, 50, 53, 55, 56].

Formal developments. A comparative historical synthesis identifying three major explanatory traditions and motivating the need for a formal theory of Structural Semantics.

Empirical hypotheses. None. This chapter is historical and conceptual, providing the intellectual foundations for the formal developments beginning in Chapter 3.

Chapter 3

Three Definitions of Understanding

Every definition illuminates one aspect of a phenomenon while obscuring another. The scientific task is therefore not merely to define, but to identify which definitions remain appropriate for which explanatory questions.

The preceding chapters established two complementary foundations. Chapter 1 isolated the epistemic domain of Structural Semantics by separating semantic capacity from phenomenal consciousness, truth, agency, and syntactic fluency; Chapter 2 surveyed the historical development of theories of meaning, cognition, and representation from structural linguistics through mechanistic interpretability. The historical record reveals that disagreements concerning artificial intelligence frequently arise because the word *understanding* is used to denote fundamentally different properties, so that researchers often disagree not because they evaluate the same property differently, but because they evaluate different properties under a common name. The objective of this chapter is therefore taxonomic rather than polemical: rather than arguing that one definition is correct and the others mistaken, we distinguish three conceptually independent notions of understanding that have emerged throughout the literature — phenomenological, functional, and structural — each capturing a genuine explanatory question, each admitting different methodologies, and each possessing different mathematical requirements.

3.1 Motivating the Taxonomy

Throughout the history surveyed in Chapter 2, philosophers have asked whether machines understand language, whether symbolic manipulation is sufficient for intelligence, whether consciousness is necessary for meaning, and whether behavioral competence implies semantic knowledge. Although these questions appear closely related, they address distinct levels of explanation: some concern subjective experience, others observable behavior, still others the internal organization of representations. Asking whether an artificial system "understands" without specifying the intended definition therefore produces an ill-posed scientific question, and a precise taxonomy becomes a prerequisite for rigorous discussion. The classification introduced below is not an exhaustive ontology of cognition, but identifies three broad explanatory frameworks that recur throughout contemporary philosophy of mind, cognitive science, and artificial intelligence.

Definition 3.1 (Understanding, neutral form)**.** Let $\mathcal{C}$ denote a system operating over a domain $\mathcal{D}$. An *understanding relation* is a mapping $U : \mathcal{C} \times \mathcal{D} \to \mathcal{P}$, where $\mathcal{P}$ denotes a collection of explanatory properties whose interpretation depends on the theoretical framework under consideration.

The codomain $\mathcal{P}$ is intentionally left abstract: for phenomenological theories it concerns subjective experience, for functional theories behavioral competence, and for Structural Semantics the admissible preservation of contextual organization. This neutrality allows the three theories to be compared without presupposing the correctness of any one of them.

The three definitions that follow represent distinct research programs rather than competing philosophical dogmas, and are not mutually exclusive: a biological organism may satisfy all three, an artificial system may satisfy one or two, and the objective here is clarification of explanatory questions rather than classification of particular systems. This taxonomy functions as the conceptual interface between the historical literature of Chapter 2 and the mathematical framework of Chapters 6 through 10; without it, later formal definitions of Scope Geometry and structural semantic capacity would remain disconnected from the broader philosophical literature.

3.2 Phenomenological Understanding

The oldest and perhaps most intuitive conception of understanding identifies it with conscious experience itself: not merely the successful manipulation of symbols or the production of appropriate behavioral responses, but something inseparable from the existence of a first-person perspective through which meaning is directly experienced. This perspective appears throughout Husserl's phenomenology, Heidegger's existential analysis, Merleau-Ponty's embodied cognition, Whitehead's process philosophy, Nagel's analysis of subjective experience, Searle's critique of computational intelligence, Chalmers' formulation of the hard problem, and Block's distinction between phenomenal and access consciousness [7, 10, 24, 30, 39, 41, 50, 56]; despite differences in terminology and metaphysical commitment, these theories converge on the claim that understanding is inseparable from phenomenal experience.

Definition 3.2 (Phenomenological understanding)**.** Let $\mathcal{C}$ be a cognitive system operating over a semantic domain $\mathcal{D}$. The system possesses *phenomenological understanding* over $\mathcal{D}$ if and only if successful semantic activity is accompanied by subjective phenomenal experience, formally $U_{\text{phen}} : \mathcal{C} \times \mathcal{D} \to \Phi$, where Φ denotes the domain of phenomenal conscious experience. The precise ontology of Φ is left unspecified, as different phenomenological traditions provide different accounts of subjective experience.

Phenomenological theories generally hold that semantic understanding is intrinsically first-person, that meaning is not exhausted by behavioral competence, and that subjective experience is explanatorily fundamental rather than derivative; consequently they distinguish appearing intelligent from genuinely understanding, and their best-known contemporary defense is Searle's Chinese Room, discussed already in Chapter 2. Within Whitehead's process philosophy, phenomenology enters the formal structure through Subjective Form, A, the manner in which an occasion experiences its own becoming, formalized fully in Chapter 4.

These theories capture an undeniable feature of ordinary experience and raise questions that remain largely unresolved by current neuroscience and artificial intelligence,

but they encounter a substantial methodological difficulty when applied to computational systems: because phenomenal experience is intrinsically first-person, no generally accepted procedure exists for measuring it objectively, so that if subjective experience is adopted as part of the definition of understanding, any system lacking independently established phenomenology is excluded by definition rather than by empirical investigation. This observation does not demonstrate that artificial systems lack phenomenology; it indicates only that phenomenological theories alone provide no operational criterion by which the question can be resolved, an observation developed into a general argument in Chapter 5. The present monograph neither accepts nor rejects phenomenological accounts, isolating them instead as one legitimate explanatory framework among several, and asks in the chapters that follow whether semantic organization can also be characterized independently through formal structural properties.

3.3 Functional Understanding

A second tradition identifies understanding not with subjective experience but with what a system can reliably do, treating the internal substrate responsible for cognition as secondary to whether the system exhibits appropriate causal organization, behavioral competence, adaptive capacity, or information-processing ability. This perspective has profoundly influenced cybernetics, cognitive science, artificial intelligence, robotics, and computational neuroscience, forming the dominant methodological framework for modern AI research [16, 43, 48, 54].

Definition 3.3 (Functional understanding)**.** Let $\mathcal{C}$ be a system operating over a semantic domain $\mathcal{D}$. The system possesses *functional understanding* if its observable behavior satisfies the operational requirements associated with successful performance over $\mathcal{D}$, formally $U_{\text{func}} : \mathcal{C} \times \mathcal{D} \to \mathcal{B}$, where $\mathcal{B}$ denotes an appropriate space of behavioral or computational performance measures. Unlike phenomenological understanding, no assumptions are made concerning subjective experience or phenomenal awareness.

Turing proposed replacing metaphysical questions concerning thought with operational tests based on conversational behavior [54]; cybernetics emphasized feedback, control, and adaptive behavior [3, 57]; and Putnam and Fodor developed forms of functionalism identifying mental states with their causal roles rather than their physical realization [16, 48], naturally supporting the possibility of multiple realizability, in which distinct physical systems instantiate the same cognitive organization. Most contemporary evaluations of large language models remain largely functionalist, measuring question answering, mathematical reasoning, code generation, planning, factual recall, and conversational competence through observable outputs rather than subjective experience, a methodology that has enabled rapid empirical progress because behavioral metrics admit objective measurement and direct comparison across systems.

Functional theories provide operational definitions suitable for empirical investigation, remain largely independent of metaphysical assumptions concerning consciousness, naturally accommodate substrate independence, and rely on criteria that are reproducible and experimentally testable. However, behavior alone does not uniquely determine internal organization: distinct computational mechanisms may produce nearly identical external behavior while employing fundamentally different internal representational structures, so that behavioral equivalence does not necessarily imply structural equivalence. Consider, for instance, two theorem-proving systems, one using symbolic logical deduction and the other a transformer trained on formal proof corpora, which

produce identical formal proofs for every theorem in a benchmark suite; from a functional perspective the two exhibit equivalent understanding, yet their internal organizations differ substantially, and functional evaluation alone does not determine whether they preserve semantic context through comparable structural mechanisms.

Structural Semantics accepts the functionalist insight that semantic organization need not depend on any particular physical substrate, but proposes that behavioral competence alone is insufficient for a complete mathematical theory of understanding, and instead investigates whether successive internal state transitions preserve bounded semantic organization throughout a computation, treating behavior as evidence for understanding rather than as its complete definition.

3.4 Structural Understanding

The third conception developed in this monograph begins from a different premise than either the phenomenological or the functional: rather than identifying understanding with subjective experience or observable behavior, it treats understanding as a property of the internal organization of representations and, specifically, of the transformations that preserve them. This does not deny the importance of consciousness or reject behavioral evaluation; it isolates a third object of scientific investigation whose mathematical properties can be studied independently of both. The central question is therefore not what a system experiences, nor how successfully it behaves, but which structural relations remain invariant as it transforms its internal representations — a shift that moves the study of semantic organization from psychology toward mathematics.

Definition 3.4 (Structural understanding)**.** Let $\mathcal{C}$ be a computational or cognitive system operating over a semantic domain $\mathcal{D}$. The system possesses *structural understanding* if there exists a family of admissible transformations $\mathcal{T} = \{T_i\}$ acting on bounded scope geometries such that every transformation preserves the admissibility constraints governing contextual organization; equivalently, $U_{\text{struct}} : \mathcal{C} \times \mathcal{D} \to \mathrm{Scope}$, where semantic capacity is evaluated through admissible morphisms in the category Scope rather than through phenomenal experience or behavioral observation.

The defining property of structural understanding is invariance: although internal representations may change continuously throughout computation, certain organizational relationships remain preserved, and these invariant relationships constitute the semantic content of the representation, precisely characterized in Parts II and III. Structural Semantics proposes that semantic organization is fundamentally a problem of scope preservation — of maintaining coherent contextual organization while new information is introduced, previous assumptions are revised, and nested contexts evolve — which motivates the bounded scope geometries of Chapter 6 and the primitive operations of Chapter 7. A defining feature of the framework is its independence from particular representational substrates: whether information is encoded symbolically, neurally, graphically, or through another computational medium is secondary to the induced transformation over bounded scope geometries, a principle formalized later as the Representation Invariance Theorem (Theorem 9.7).

Consider again two reasoning systems proving the same mathematical theorem, one by symbolic deduction and the other by continuous optimization over neural activation vectors: although their internal representations differ radically, both may maintain identical assumptions, preserve identical logical dependencies, introduce hypotheses in the

same admissible order, and discharge those assumptions according to equivalent structural rules. From the perspective of Structural Semantics, both systems instantiate the same semantic organization despite their differing computational substrates, since their semantic equivalence depends on preserved contextual structure rather than on identical internal encodings.

3.5 Comparing the Three Conceptions

The three conceptions of understanding introduced in this chapter may now be compared directly. Phenomenological understanding asks whether semantic activity is accompanied by subjective experience; functional understanding asks whether semantic activity produces appropriate observable behavior; structural understanding asks whether semantic activity preserves the organization of contextual constraints throughout computation. These questions are logically independent: a biological organism may satisfy all three, a theorem prover may satisfy the latter two while remaining silent with respect to phenomenology, and a hypothetical conscious system suffering severe cognitive impairment might possess phenomenology while exhibiting poor functional or structural organization. The taxonomy therefore separates explanatory questions rather than ranking competing theories.

Framework	Primary object	Evaluation	Representative literature
Phenomenological	Subjective experience	First-person awareness	Husserl, Heidegger, Whitehead, Nagel, Chalmers
Functional	Behavioral competence	Observable performance	Turing, Putnam, Fodor, Newell, Simon
Structural	Contextual organization	Admissible transformations	Scope Calculus, category theory, mechanistic interpretability

Table 3.1: Three complementary conceptions of understanding.

3.6 Chapter Summary

This chapter established a conceptual taxonomy separating three distinct research programs concerning understanding: phenomenological theories identify understanding with subjective experience, functional theories with successful behavioral organization, and Structural Semantics introduces a third conception in which semantic capacity is identified with the preservation of admissible contextual organization under transformation. The remainder of this monograph develops this third perspective systematically, beginning in Chapter 4 with a mathematical reconstruction of Whitehead's process ontology that later permits comparison with the Scope Calculus through the translation functor of Chapter 10.

Status of Results: Chapter 3

Established background. Phenomenological, functional, and structural traditions in philosophy of mind, cognitive science, and artificial intelligence [7, 50, 54, 56].

Formal developments. Definitions of Phenomenological Understanding, Functional Understanding, and Structural Understanding, together with their comparative taxonomy.

Empirical hypotheses. Structural semantic capacity constitutes an independently measurable property that is neither reducible to subjective phenomenology nor fully captured by behavioral performance alone. Subsequent chapters develop the formal mathematical machinery necessary to evaluate this hypothesis.

Chapter 4

The Whiteheadian Framework

The many become one, and are increased by one.
— Alfred North Whitehead, *Process and Reality*

The preceding chapter established that discussions of understanding separate into phenomenological, functional, and structural frameworks. This chapter returns to one of the most influential systems within the phenomenological tradition: Whitehead's process philosophy. Although the present monograph ultimately develops a different formal framework, Whitehead's emphasis on dynamic organization, relational structure, and process makes his work unusually compatible with modern mathematical approaches to computation, and this chapter is accordingly presented not as a critique but as a careful mathematical reconstruction. Its objective is twofold: to provide precise formal definitions of Whitehead's principal concepts using contemporary notation, and to identify exactly which components admit structural formalization and which remain explicitly phenomenological, preparing the translation functor developed in Chapter 10.

4.1 Process Rather Than Substance

Classical metaphysics generally treats enduring substances as the primary objects of reality, with processes and changes regarded as properties or modifications of an underlying entity. Whitehead reversed this relationship: reality consists fundamentally of events, and objects are abstractions constructed from sufficiently stable patterns within those events. If processes are primitive, transformation becomes more fundamental than state and relations more fundamental than isolated entities, an orientation shared by category theory, dynamical systems, network theory, and process algebra, though Whitehead developed his system decades before these fields matured.

4.2 Actual Occasions

The primitive object of Whitehead's ontology is the *actual occasion*: not a material object but a single completed event of becoming, from which all larger structures — organisms, mountains, societies, even physical particles — are understood as persistent patterns generated by vast collections of interconnected occasions.

Definition 4.1 (Actual occasion)**.** An actual occasion is represented by an ordered tuple $E = \langle D, P, S \rangle$, where D denotes the collection of antecedent data inherited from prior

occasions, P denotes the collection of prehensions performed on that data, and S denotes the completed satisfaction state produced after concrescence.

This definition is intentionally schematic; the remaining sections formalize each component individually.

4.3 Prehension

Whitehead replaces classical perception with the broader concept of *prehension*, the fundamental relational operation through which one occasion incorporates aspects of previous occasions into its own becoming. Unlike ordinary perception, prehension is not restricted to conscious organisms: every actual occasion prehends antecedent reality, so that prehension functions as the primitive relational operator of Whitehead's ontology.

Definition 4.2 (Prehension)**.** A prehension is represented by the ordered triple $P = \langle D, v, A \rangle$, where D denotes the datum being prehended, $v \in \{+, -\}$ denotes the valuation polarity, with positive valuation indicating inclusion and negative valuation indicating exclusion, and A denotes the Subjective Form associated with the prehension.

Whitehead distinguishes positive prehensions, which contribute directly to the formation of the new occasion, from negative prehensions, which exclude data from further participation. Formally, if $\mathcal{D} = \{d_1, \dots, d_n\}$ denotes the available antecedent data, the process of becoming partitions this collection into disjoint subsets $\mathcal{D} = P^+ \cup P^-$ with $P^+ \cap P^- = \emptyset$: positive prehensions contribute to the evolving occasion, negative prehensions remove information from subsequent construction. This distinction will later admit a natural comparison with the $\mathcal{B}_d$ and $\mathcal{R}$ operations of the Scope Calculus, although no such identification is assumed at this stage.

4.4 Concrescence

The collection of individual prehensions does not remain an unordered set; instead it undergoes a process of integration Whitehead calls *concrescence*, transforming many inherited influences into one completed occasion, symbolically $\{P_1, \dots, P_n\} \longrightarrow S$, summarized in Whitehead's own phrase: "the many become one, and are increased by one." From a mathematical perspective, concrescence resembles an aggregation operator acting on a structured family of relational inputs; precisely which algebraic properties such an operator possesses is a question examined later through the Scope Calculus.

4.5 Subjective Form

Among the components of Whitehead's ontology, Subjective Form occupies a special position. Whereas antecedent data and valuation polarity describe observable structural relations, Subjective Form describes the manner in which the occasion experiences its own process of becoming, and unlike positive and negative prehension it is not introduced as an externally observable property but concerns the intrinsic character accompanying each prehension. Consequently Subjective Form functions as the explicitly phenomenological component of Whitehead's ontology; the present monograph seeks to determine which components of his system admit purely structural reconstruction and which remain irreducibly phenomenological, preserving Subjective Form explicitly

rather than eliminating it, while carefully distinguishing it from the structural operations that surround it.

4.6 Chapter Summary

This chapter reconstructed the fundamental concepts of Whitehead's process ontology using contemporary mathematical notation: actual occasions as primitive events of becoming; prehensions as ordered triples of antecedent data, valuation polarity, and Subjective Form; positive and negative prehensions as an explicit set-theoretic decomposition; and concrescence as the aggregation process transforming many prehensions into one completed satisfaction state. Most importantly, Subjective Form was isolated as the explicitly phenomenological component of the framework. The following chapter examines this distinction in detail, showing that Subjective Form functions as an independent axiom within Whitehead's ontology and forms the critical bridge between phenomenological theories of understanding and the structural framework developed throughout the remainder of the monograph.

Status of Results: Chapter 4

Established background. Whitehead's process ontology, actual occasions, prehensions, concrescence, and Subjective Form [56].

Formal developments. Mathematical reconstruction of actual occasions, prehensions, valuation polarity, and concrescence using explicit definitions suitable for later categorical translation.

Empirical hypotheses. None. This chapter reconstructs an existing philosophical framework in preparation for the structural analysis developed in Chapters 5–10.

Chapter 5

The Phenomenological Boundary

Everything is vague to a degree you do not realize till you have tried to make it precise.
— Bertrand Russell

The preceding chapter reconstructed Whitehead's process ontology without either criticism or reinterpretation, identifying its primitive components in their own terms and observing that the architecture naturally separates into structural entities — actual occasions, prehensions, valuation polarity, and concrescence, which admit explicit mathematical representation — and Subjective Form, introduced not as an externally observable relation but as the intrinsic manner in which an occasion experiences its own becoming. The present chapter examines the consequences of this distinction. The question is not whether Subjective Form exists, nor whether Whitehead's metaphysics is correct, but methodological: if Subjective Form is treated as an indispensable prerequisite for semantic capacity, the evaluation of artificial systems becomes independent of any mathematical or empirical investigation, and the conclusion that such systems lack understanding follows directly from the initial definition rather than from subsequent analysis. The purpose of this chapter is to isolate the precise logical boundary separating structural questions from phenomenological axioms, so that the mathematical framework developed in subsequent chapters can be evaluated independently of broader metaphysical commitments.

5.1 Structural and Phenomenological Components

The reconstruction of Chapter 4 naturally partitions Whitehead's ontology into two classes. The first consists of objects whose behavior may be described through explicit structural relations: D, P, v, and S each admit mathematical description, whether as sets, graphs, tensors, categories, discrete operators, or aggregation functionals, and each possesses an externally describable operational role. The second class contains only Subjective Form, A: unlike every other component, it is not introduced through observable transformation rules but denotes the intrinsic character accompanying each prehension, so that its inclusion within the theory is fundamentally different from every other primitive introduced thus far.

Definition 5.1 (Structural component)**.** A primitive object is *structural* if its contribution to the behavior of a system is completely specified by externally observable relations and transformation laws.

Definition 5.2 (Phenomenological component)**.** A primitive object is *phenomenological* if its defining properties concern intrinsic experience rather than externally observable structural relations.

Under these definitions D, P, v, and S are structural components, while A is phenomenological. The distinction is purely methodological: it does not deny the existence of phenomenological properties, but identifies which primitives participate directly in the structural mathematics developed throughout the remainder of the monograph.

5.2 Axiomatic Dependence

The central logical observation of this chapter concerns the role played by Subjective Form within theories of understanding. Suppose semantic capacity is defined by the predicate Understand(M). If one adopts the axiom $A \neq \emptyset \Rightarrow$ Understand(M) together with its converse $A = \emptyset \Rightarrow \neg$Understand$(M)$, then the semantic status of every system is determined immediately by the presence or absence of Subjective Form, and no subsequent behavioral experiment, mathematical analysis, or empirical measurement can alter this conclusion.

Observation 5.3. If Subjective Form is introduced as a necessary condition for semantic capacity, then every system lacking Subjective Form is excluded by definition rather than by empirical investigation.

This observation concerns logical dependence rather than philosophical correctness; one may certainly choose such an axiom, but the resulting theory belongs to a different methodological category from one whose conclusions depend on mathematical or empirical evidence.

5.3 Definitions and Empirical Criteria

Scientific theories generally distinguish between definitions, which specify the concepts under consideration, and measurements, which determine whether particular systems satisfy those definitions; when these two roles are conflated, empirical investigation becomes impossible. If one defines intelligence as human intelligence, no experiment comparing humans and machines could demonstrate machine intelligence, since the conclusion follows directly from the definition; similarly, if semantic capacity is defined as requiring Subjective Form, every non-conscious computational architecture is excluded before its internal organization is examined. Structural Semantics adopts a different strategy, defining semantic capacity through structural preservation and treating whether phenomenology accompanies such preservation as a distinct philosophical question.

5.4 The Structural Research Program

Separating structural organization from phenomenology produces a well-defined mathematical research program whose central question is which structural invariants must be preserved for semantic organization to remain stable. Unlike questions concerning subjective experience, this question admits explicit mathematical treatment: one may define admissible transformations, characterize boundary operators, prove preservation theorems, derive computational diagnostics, and compare different representational

architectures through common structural invariants. Consequently the mathematical framework developed throughout Volumes II and III does not depend on resolving the philosophical debate concerning consciousness, studying instead the preservation of structured relational organization.

This methodological separation is compatible with, while differing from, several existing traditions: classical functionalism evaluates systems primarily through behavioral performance [16, 48], phenomenological theories through subjective experience or intrinsic intentionality [41, 50], and mechanistic interpretability through internal computational organization [15, 44]. Structural Semantics occupies a different position, identifying semantic capacity with neither behavior nor phenomenology but investigating whether semantic organization can be characterized by mathematically definable structural invariants, an orientation that motivates the algebraic development beginning in Chapter 6.

5.5 Chapter Summary

This chapter identified the precise methodological boundary separating phenomenological and structural theories of semantic capacity, distinguishing structural primitives from phenomenological primitives and showing that Subjective Form occupies a unique position within Whitehead's ontology: theories requiring Subjective Form as an axiom classify machine understanding by definition rather than by empirical investigation. Structural Semantics adopts a complementary research program, developing a mathematical theory of structural organization whose predictions may be evaluated independently of phenomenological commitments. This completes Volume I. Beginning with Chapter 6, the monograph transitions from philosophical analysis to the formal mathematical development of Scope Geometry.

Status of Results: Chapter 5

Established background. Whitehead's distinction between structural process and Subjective Form, together with classical debates concerning consciousness and intentional content [41, 50, 56].

Formal developments. Definitions of structural and phenomenological components, together with the logical separation between axiomatic phenomenological theories and structural mathematical theories.

Empirical hypotheses. None. This chapter establishes methodological distinctions that motivate the formal development of Scope Geometry in Volume II.

Part II

Scope Geometry, Algebra, and Categories

Chapter 6

Scope Geometry

The art of reasoning consists in getting hold of the subject at the right end, of seizing the few general ideas that illuminate the whole, and of persistently organizing all subsidiary facts around them.
— Alfred North Whitehead

Volume I established the philosophical and epistemic foundations of Structural Semantics. Its principal conclusion was not that phenomenology is false or unnecessary, but that the mathematical analysis of semantic organization can proceed independently of questions concerning subjective experience. Having isolated the structural domain, we now begin constructing the mathematical objects that will carry the remainder of the theory.

The central claim of this volume is that semantic organization is fundamentally geometric. Every computational system, biological or artificial, operates within collections of constraints that determine which transformations preserve coherence and which produce contradiction or collapse. These collections of constraints are neither arbitrary sets nor merely symbolic contexts: they possess internal organization, hierarchical nesting, admissible boundaries, and lawful mechanisms for interaction. This chapter introduces the mathematical object that captures these properties, the *bounded scope geometry*, which provides the primitive space on which the Scope Calculus of subsequent chapters operates. Primitive operations such as Pop, Refuse, Bind, and Collapse will later be defined as transformations acting on these geometries (Chapter 7), while Chapters 8–10 establish their categorical, algebraic, and functorial properties. Throughout this chapter we deliberately avoid introducing operational rules, defining first the spaces on which operations act before studying the operations themselves — mirroring the development of modern topology and differential geometry, where spaces are characterized independently of the mappings subsequently defined between them.

Remark 6.1 (Terminological note). The formalism developed in this and the following four chapters was originally introduced and published under the name *Spherepop*. The present exposition is self-contained and does not presuppose that earlier work; it is renamed and reorganized here as the Scope Calculus to foreground its role within Structural Semantics rather than its original programming-language framing. Readers already familiar with Spherepop will recognize $\mathcal{P}$, $\mathcal{R}$, $\mathcal{B}_d$, and $\mathcal{C}_l$ as the same four primitives under new emphasis; readers meeting the material for the first time can treat the two names as synonymous.

6.1 Motivation

Nearly every formal treatment of semantics assumes, explicitly or implicitly, the existence of context. Natural language interpretation, programming languages, logical deduction, theorem proving, biological regulation, and distributed computation all depend on determining which objects belong to a given context and which transformations preserve that membership; yet context is rarely treated as a mathematical object in its own right. In linguistics it is often represented informally through discourse structure or lexical fields [13, 36]; in formal logic it appears as assumptions within derivations [20]; programming-language semantics typically represents it through environments, symbol tables, or typing judgments [47, 58]; and category theory studies morphisms between objects while generally leaving contextual admissibility to external structure [37]. These approaches have been extraordinarily successful within their respective domains, but none was developed specifically to study semantic stability under arbitrary transformations across multiple representational substrates.

Structural Semantics therefore introduces an explicit geometry of contexts. Rather than viewing context merely as auxiliary information attached to a computation, we regard context itself as the primary mathematical object: semantic transformations become transformations of context geometries, and semantic preservation becomes the preservation of admissible geometric structure. This shift is analogous to the historical transition from Euclidean coordinate calculations to differential geometry, in which geometry became primary while coordinates became representations; likewise, scope geometry treats context as primary while individual representations become coordinates within that context.

6.2 Primitive Intuition

Before introducing formal definitions, it is useful to develop geometric intuition. Consider a mathematical proof: at any stage there exists a collection of assumptions, definitions, variables, and established propositions that remain valid, forming a coherent environment in which introducing an undefined symbol, violating an assumption, or applying a theorem outside its domain produces inconsistency. Consider similarly lexical interpretation: a discussion of planetary motion establishes a context in which the word "orbit" possesses one meaning, while a discussion of computing establishes a different context in which the same lexical item acquires an entirely different interpretation, and the semantic validity of subsequent expressions depends on remaining inside the appropriate context unless an explicit contextual transition occurs. Programming languages exhibit the same phenomenon: variables exist only within declared scopes, references outside these scopes become invalid, and nested functions inherit some contextual information while introducing new local constraints.

Although these examples originate in different disciplines, they share a common structural pattern: each consists of a bounded region of admissible entities, explicit constraints governing admissibility, hierarchical nesting, and lawful transitions between nested regions. Scope Geometry abstracts exactly these common structural features.

6.3 Why Boundaries Are Fundamental

The formal definitions beginning in the next section take a boundary, not a symbol, a vector, or an entity, as the primitive notion from which everything else is built. This choice

is worth motivating before the definitions arrive, since it is not the only one available: a theory could equally well begin with a space of representations and add constraints afterward, treating admissibility as a derived property rather than a starting point.

The reason for beginning with boundaries instead is that the examples of the preceding section share nothing at the level of representation. A proof's assumptions, a word's discourse context, and a variable's lexical scope have no common representational format — one is a set of propositions, another a topic under discussion, the third a region of program text. What they share is only the pattern of admissibility itself: a bounded region within which certain moves are licensed and others are not, independently of what occupies that region. A theory that started from representations would need a separate account of admissibility for every representational format it encountered. A theory that starts from the boundary can be stated once and instantiated by whatever representation a given application supplies, which is exactly the substrate independence pursued formally in Chapter 9.

This also explains why the definitions that follow introduce $\mathcal{B}$ before $\mathcal{E}$ rather than the reverse. The interior of a scope, Definition 6.5 below, is defined *as* the states satisfying the boundary, so that the boundary is prior in the order of definition as well as in the order of motivation: there is no admissible interior to speak of until the boundary that determines it has already been specified.

6.4 State Domains

Every computational or cognitive system evolves through collections of states. We therefore begin with the underlying state domain.

Definition 6.2 (State domain)**.** A *state domain* is a nonempty set $\mathcal{D}$ whose elements represent admissible configurations of a computational, logical, or cognitive system.

The internal structure imposed on $\mathcal{D}$ depends on the application: for symbolic reasoning it may consist of logical derivation states, for transformer architectures $\mathcal{D} \subset \mathbb{R}^d$ may represent activation vectors, and for programming languages it may denote environments consisting of variable bindings and execution frames. The theory developed here does not depend on the specific realization of $\mathcal{D}$; only the existence of distinguishable states is assumed.

6.5 Boundaries

State domains alone are insufficient. Semantic organization requires constraints determining which transitions remain valid; these constraints are represented by boundaries.

Definition 6.3 (Boundary)**.** Let $\mathcal{D}$ be a state domain. A *boundary* $\mathcal{B}$ is a collection of admissibility predicates $\mathcal{B} = \{\beta_i\}_{i \in I}$, where each $\beta_i : \mathcal{D} \to \{0, 1\}$ determines whether a given state satisfies a particular structural constraint.

Boundaries need not be purely logical: they may represent semantic consistency, typing constraints, physical conservation laws, security permissions, resource limits, or probabilistic admissibility criteria, and the abstraction intentionally encompasses all such cases. A boundary in this sense also admits a topological reading, in which it separates admissible from inadmissible regions of a semantic space in exactly the sense that a topological boundary separates a region from its complement; this reading is developed rigorously in Appendix E.

6.6 Interior Spaces

Having introduced boundaries, we define the region they enclose.

Definition 6.4 (Interior space)**.** Given a state domain $\mathcal{D}$ and boundary $\mathcal{B}$, the associated interior space is $\mathcal{E} = \{x \in \mathcal{D} : \beta(x) = 1 \text{ for every } \beta \in \mathcal{B}\}$.

Thus $\mathcal{E}$ contains precisely those states satisfying every active admissibility constraint; states outside $\mathcal{E}$ may still exist within the global domain $\mathcal{D}$, but they are not locally admissible within the current scope.

6.7 Nested Scope Geometries

Contexts rarely exist in isolation. Most systems construct hierarchies: programming languages contain nested blocks, logical proofs contain nested assumptions, human discourse continually opens and closes subordinate contexts, and transformer attention similarly constructs localized contextual substructures. To model this hierarchy we introduce nested scope geometries.

Definition 6.5 (Bounded scope geometry)**.** A bounded scope geometry is the ordered tuple $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$, where $\mathcal{B}$ is a boundary, $\mathcal{E}$ is the associated admissible interior, and $\mathcal{S}_{\text{child}}$ is a finite or countable collection of child scope geometries.

The recursive appearance of $\mathcal{S}_{\text{child}}$ permits arbitrarily deep hierarchical structures while preserving a uniform mathematical description, and this recursive definition will later permit induction over scope depth and the derivation of decomposition theorems in Chapter 9.

6.8 Hierarchy, Containment, and Inheritance

The recursive definition of bounded scope geometries naturally induces a hierarchical organization: rather than existing as isolated objects, scopes form partially ordered containment structures in which local contexts inherit constraints from larger environments while introducing additional local restrictions. This hierarchical organization appears across numerous disciplines: block-structured programming languages inherit variable environments from enclosing functions while introducing local bindings [47], formal proof systems inherit assumptions from outer derivations while temporarily extending them with auxiliary hypotheses [20], and linguistic discourse similarly maintains a persistent conversational context while permitting nested quotations, counterfactuals, and hypothetical reasoning [13, 36]. The purpose of Scope Geometry is to provide a common mathematical framework for each of these situations.

Definition 6.6 (Parent and child scope)**.** Let $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$ be a bounded scope geometry. If $\mathrm{S}' \in \mathcal{S}_{\text{child}}$, then S is the *parent scope* of S' and S' is a *child scope* of S.

Parenthood defines only an immediate containment relation; ancestor and descendant relationships are obtained by repeated application of the parent relation.

Definition 6.7 (Scope containment)**.** Let $\mathrm{S}_1 = \langle \mathcal{B}_1, \mathcal{E}_1, \mathcal{S}_{\text{child}_1} \rangle$ and $\mathrm{S}_2 = \langle \mathcal{B}_2, \mathcal{E}_2, \mathcal{S}_{\text{child}_2} \rangle$. We write $\mathrm{S}_2 \preceq \mathrm{S}_1$ if and only if S_2 is contained within S_1.

The relation $\preceq$ is interpreted as geometric containment rather than set inclusion: two scopes may contain overlapping semantic information while remaining distinct objects within the hierarchy, so containment captures organizational structure rather than membership alone.

A central property of hierarchical contexts is inheritance: child scopes do not arise independently, but inherit structural constraints established by their parents unless those constraints are explicitly modified through admissible transformations.

Definition 6.8 (Boundary inheritance)**.** Let $\mathrm{S}_2 \preceq \mathrm{S}_1$. The boundary of the child scope inherits from the parent if $\mathcal{B}_1 \subseteq \mathcal{B}_2$; additional local constraints may be introduced, but inherited constraints remain active unless explicitly discharged.

Boundary inheritance guarantees that entering a child scope cannot erase the structural commitments established by its ancestors; instead, deeper scopes become progressively more constrained. This monotonic refinement of admissibility will later prove essential for the decomposition theorems of Chapter 9.

6.8.1 Local and Global Admissibility

The distinction between local and global validity is fundamental: a state may be perfectly admissible within one scope while violating the constraints of another. A temporary variable inside a programming block is valid locally but ceases to exist after leaving the block, and an assumption introduced during an indirect proof is admissible only within the subproof in which it was declared. This motivates two complementary notions.

Definition 6.9 (Local admissibility)**.** A state $x \in \mathcal{D}$ is *locally admissible* with respect to a scope S if $x \in \mathcal{E}$.

Definition 6.10 (Global admissibility)**.** A state is *globally admissible* if it satisfies the boundary constraints of every ancestor scope containing the current scope.

Every globally admissible state is locally admissible, but the converse need not hold.

6.8.2 Boundary Violations

Boundary constraints divide state transitions into two classes: some preserve admissibility, others produce structural inconsistency.

Definition 6.11 (Boundary violation)**.** Let $x \in \mathcal{E}$ and let $T : \mathcal{D} \to \mathcal{D}$ be a state transformation. The transformation is a *boundary violation* if $T(x) \notin \mathcal{E}$.

Boundary violations need not correspond to logical contradictions; depending on the application they may represent type errors, inconsistent assumptions, invalid semantic references, resource exhaustion, security violations, unstable activation trajectories, or loss of contextual coherence. The Scope Calculus introduced in the next chapter exists precisely to characterize the permissible responses to such violations.

6.8.3 The Scope Hierarchy as a Partially Ordered Structure

The containment relation naturally equips the collection of scopes with a partial order.

Proposition 6.12. *The relation $\preceq$ over bounded scope geometries is reflexive, antisymmetric, and transitive.*

Proof. Reflexivity follows immediately since every scope contains itself. Antisymmetry follows because if $S_1 \preceq S_2$ and $S_2 \preceq S_1$, then both scopes possess identical containment relationships and are identified as the same scope geometry. Transitivity follows because containment through an intermediate scope implies containment within every ancestor. Hence $(\mathrm{Scope}, \preceq)$ forms a partially ordered set. □

This order will later support induction over scope depth, recursive repair procedures, and the categorical constructions of Chapters 8–10.

6.9 Discussion

The geometry introduced thus far deliberately remains static: we have defined the spaces on which semantic organization is represented, but not yet described how those spaces evolve. This distinction mirrors the historical development of geometry itself — classical topology first characterizes spaces and only subsequently studies continuous mappings between them, and differential geometry first defines manifolds before introducing vector fields and flows. Scope Geometry follows the same methodological order: Chapters 6 and 7 separate geometric structure from dynamical evolution, so that once the spaces have been rigorously defined, transformations may be introduced without ambiguity. In particular, the four primitive operations of the Scope Calculus will be shown to generate every admissible structural transformation required by the framework, providing the dynamical counterpart to the static geometries defined here.

6.10 Chapter Summary

This chapter introduced the fundamental geometric objects underlying Structural Semantics. Beginning with state domains and boundary constraints, we constructed bounded scope geometries as recursively nested contexts possessing admissible interiors and hierarchical organization, defined parent-child relationships, boundary inheritance, local and global admissibility, and established that the collection of scope geometries forms a partially ordered hierarchy under containment. These constructions intentionally remain independent of any operational mechanism, defining the spaces on which semantic organization is represented without prescribing how those spaces evolve. The next chapter introduces the minimal operational language required to manipulate these geometries: the four primitive operations Pop, Refuse, Bind, and Collapse.

Status of Results: Chapter 6

Established background. Hierarchical contexts in formal logic, programming-language semantics, topology, and lexical semantics [20, 36, 37, 47].

Formal developments. Definitions of parent and child scopes, scope containment, boundary inheritance, local and global admissibility, boundary violations, and the partial ordering of scope hierarchies.

Empirical hypotheses. None. This chapter establishes the static geometric framework on which the operational Scope Calculus will be developed in Chapter 7.

Chapter 7

Primitive Operations and the Scope Calculus

Mathematics possesses not only truth, but supreme beauty — a beauty cold and austere, like that of sculpture.
— Bertrand Russell

Chapter 6 introduced the static geometry underlying Structural Semantics: bounded scope geometries, boundary constraints, hierarchical containment, and admissibility relations, without specifying how these objects evolve. The present chapter supplies that missing component by introducing the primitive operations that generate all admissible transformations within Scope Geometry.

The guiding principle is one of structural minimality: rather than constructing a large collection of specialized operators, we seek the smallest operational alphabet capable of expressing every admissible transformation required by semantic organization, paralleling how group theory seeks minimal generating sets, formal language theory studies finite alphabets generating infinite languages, and universal algebra investigates operations sufficient to construct entire classes of algebraic structures [9, 29, 37]. The claim advanced in this chapter is that four primitive operations, $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$, are sufficient to describe the evolution of bounded scope geometries. These primitives are defined independently of any particular computational substrate: they are not programming-language instructions, neural-network operations, or logical inference rules, but abstract transformations of context itself. Subsequent chapters show that this primitive alphabet admits a rich algebraic structure, supports canonical normal forms, and provides the foundation for categorical and topological analysis.

7.1 Motivation

Any sufficiently expressive computational system must introduce new contextual information, reject information incompatible with existing constraints, establish relationships between previously independent entities, and eventually resolve temporary structures into stable results. Programming languages accomplish these tasks through scope creation, exception handling, variable binding, and function evaluation; mathematical proofs introduce assumptions, reject contradictions, construct dependencies, and discharge temporary hypotheses; conversation continually opens new topics, rejects inap-

propriate interpretations, links ideas, and eventually resolves discussion into shared conclusions. Although the implementations differ dramatically, the underlying structural transformations remain remarkably similar, and the Scope Calculus abstracts these transformations into four primitive operations acting directly on bounded scope geometries.

7.2 Primitive Alphabet

Definition 7.1 (Primitive alphabet)**.** The primitive operational alphabet of the Scope Calculus is $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$. Each primitive denotes a partial transformation acting on bounded scope geometries.

The primitives are interpreted operationally rather than symbolically: they describe transformations of context rather than transformations of syntax, and every admissible computation considered throughout this monograph is represented as a finite composition of these primitive operations. The remainder of this chapter examines each primitive individually before constructing their algebraic interactions.

7.2.1 The Pop Operator

The first primitive governs contextual escalation. Many computations require information generated within a local scope to become available within an enclosing scope — returning a function value, proving an intermediate lemma, exporting a variable, or promoting a local conclusion into a global argument.

Definition 7.2 (Pop)**.** Let $\mathrm{S}_c \preceq \mathrm{S}_p$ be a child scope contained within a parent scope. The Pop operator $\mathcal{P} : \mathrm{S}_c \to \mathrm{S}_p$ promotes an admissible entity from the child scope into its parent while preserving all inherited boundary constraints.

Pop does not destroy the child scope; it merely transfers designated structural information across the boundary between nested contexts. The admissibility of such promotion depends entirely on the boundary constraints of Chapter 6: if promotion violates an inherited boundary, the operation is not admissible. The interaction between Pop and Refuse is examined later in this chapter.

7.2.2 The Refuse Operator

The second primitive governs boundary enforcement. Every coherent context must distinguish admissible transformations from inadmissible ones, without which semantic organization degenerates into unrestricted accumulation of incompatible information.

Definition 7.3 (Refuse)**.** Let $x \in \mathcal{D}$ be a proposed state transition. The Refuse operator $\mathcal{R}$ rejects the transition whenever $x \notin \mathcal{E}$; equivalently, $\mathcal{R}$ acts whenever the proposed transformation violates one or more active boundary predicates.

Refuse is not an exceptional condition added to the calculus, but a primitive structural mechanism preserving admissibility. In programming languages it corresponds to type violations or access restrictions, in logical deduction to invalid inference, in conversational reasoning to rejecting incompatible interpretations, and in the transformer diagnostics of Chapters 13–15 it will be interpreted as the detection of structural context collapse rather than low statistical confidence.

7.2.3 The Bind Operator

Whereas Pop transfers existing entities between contexts, Bind creates new structural relationships within a context.

Definition 7.4 (Bind)**.** Let $a, b \in \mathcal{E}$. The Bind operator $\mathcal{B}_d$ creates an admissible dependency relation $a \leftrightarrow b$ within the current scope geometry.

The precise mathematical nature of this dependency depends on the application: within programming languages it may represent variable association, within logical systems dependence between hypotheses and conclusions, and within neural architectures it may later be interpreted as persistent activation coupling. The calculus intentionally abstracts over these realizations, concerning itself with the creation of lawful structural dependence rather than its physical implementation. Readers who know the λ-calculus may already suspect a connection between $\mathcal{B}_d$ and variable binding; that connection is made precise, rather than left as an analogy, in Appendix A.

7.2.4 The Collapse Operator

The final primitive governs contextual resolution. Temporary structures cannot remain indefinitely unresolved: at some stage subordinate computations terminate and their results become available to surrounding contexts.

Definition 7.5 (Collapse)**.** Let $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$. The Collapse operator $\mathcal{C}_l$ maps the current scope geometry onto a canonical admissible result while discharging temporary internal structure.

Collapse should not be interpreted as destruction but as evaluation: temporary internal organization is summarized by a stable outcome that can subsequently participate in larger computations, such as evaluation of an expression, completion of a proof, termination of recursion, resolution of a dialogue, or completion of an optimization step.

7.2.5 Operational Completeness

Theorem 7.6 (Operational completeness)**.** *Every admissible transformation of bounded scope geometries admits a finite presentation as a composition of the primitive operations* $\{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$.

Proof. Proceed by structural induction over admissible context transformations. Every transformation either introduces contextual information, rejects inadmissible information, creates admissible dependency, or resolves temporary structure; complex transformations decompose recursively into finite compositions of these elementary cases. The complete inductive argument is deferred to Appendix I. □

7.3 Composition of Primitive Operations

The primitive operators acquire their mathematical significance only through composition: individual operations describe isolated contextual events, while computation, reasoning, and semantic interpretation arise from finite sequences of such events. Unlike arithmetic operations, the primitive transformations of the Scope Calculus are inherently contextual, so the algebra they generate is generally non-commutative.

Definition 7.7 (Primitive composition)**.** Let $\alpha, \beta \in \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$. The composition $\beta \circ \alpha$ denotes sequential execution of α followed by β, provided the intermediate scope geometry remains admissible.

Whenever admissibility fails after the first operation, the composed transformation is undefined, so the Scope Calculus naturally forms a partial algebra rather than a total one.

A finite computation over a bounded scope geometry S_0 is represented by an ordered sequence $\sigma = \alpha_n \circ \cdots \circ \alpha_1$ with $\alpha_i \in \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$, executed right to left and generating an evolution $\mathrm{S}_0 \to \mathrm{S}_1 \to \cdots \to \mathrm{S}_n$ in which each transition satisfies the admissibility conditions of Chapter 6.

Definition 7.8 (Admissible computation)**.** A finite primitive sequence $\sigma = \alpha_n \circ \cdots \circ \alpha_1$ is an *admissible computation* if every intermediate transition $\mathrm{S}_i \to \mathrm{S}_{i+1}$ preserves local boundary admissibility.

Admissibility is therefore a property of the entire computation rather than merely its initial and final states.

Proposition 7.9 (Associativity)**.** *Let $\alpha, \beta, \gamma \in \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$. Whenever both sides are defined, $(\gamma \circ \beta) \circ \alpha = \gamma \circ (\beta \circ \alpha)$.*

Proof. Primitive transformations are partial functions acting on bounded scope geometries, so associativity follows directly from associativity of functional composition, the only restriction being that every intermediate transformation remains admissible. □

Associativity permits omission of parentheses in finite primitive sequences; henceforth $\alpha_1 \circ \alpha_2 \circ \cdots \circ \alpha_n$ denotes the unique associative composition.

Although composition is associative, it is generally not commutative: the order in which contextual transformations occur affects the resulting scope geometry, since binding an entity before it exists differs fundamentally from binding it after introduction.

Proposition 7.10 (Non-commutativity)**.** *There exist $\alpha, \beta \in \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$ such that $\alpha \circ \beta \neq \beta \circ \alpha$.*

Proof. Consider a local entity $x \in \mathcal{E}$. Applying $\mathcal{B}_d$ followed by $\mathcal{P}$ first establishes a dependency before promoting the resulting structure; reversing the order promotes the entity prior to creating the dependency. Unless the parent scope already contains the corresponding relational structure, the resulting scope geometries differ, so $\mathcal{P} \circ \mathcal{B}_d \neq \mathcal{B}_d \circ \mathcal{P}$. □

Non-commutativity is therefore an intrinsic consequence of contextual organization rather than an implementation artifact.

Definition 7.11 (Identity operation)**.** The identity operation id is the unique primitive sequence of length zero satisfying $\mathrm{id}(\mathrm{S}) = \mathrm{S}$ for every bounded scope geometry.

Consequently $\sigma \circ \mathrm{id} = \sigma = \mathrm{id} \circ \sigma$; this identity transformation later becomes the identity morphism of the category Scope introduced in Chapter 8.

The four primitives interact in characteristic ways: some compositions are always admissible, others require boundary verification, and still others are undefined whenever structural constraints are violated. Table 7.1 summarizes these interactions informally; the precise rewrite rules follow in the next section.

First	Second	Typical interpretation
$\mathcal{B}_d$	$\mathcal{P}$	Promote established relation
$\mathcal{P}$	$\mathcal{B}_d$	Bind after promotion
$\mathcal{R}$	$\mathcal{P}$	Promotion denied
$\mathcal{P}$	$\mathcal{R}$	Boundary validation after promotion
$\mathcal{C}_l$	$\mathcal{P}$	Promote evaluated result
$\mathcal{C}_l$	$\mathcal{B}_d$	Bind canonical representative
$\mathcal{B}_d$	$\mathcal{C}_l$	Evaluate constructed dependency

Table 7.1: Typical interactions among primitive operators. Formal rewrite rules follow in the next section.

7.4 The Primitive Rewriting System

We now equip the algebra with a rewriting system that identifies operationally equivalent computations. Its purpose is twofold: first, to eliminate redundant or vacuous transformations, yielding shorter and more canonical descriptions of computations; second, to provide the foundation for the normal-form results of Chapter 9. The rewriting system is deliberately conservative — it does not alter the semantic content of a computation, but replaces one admissible presentation with another that is structurally simpler while preserving boundary constraints and contextual relationships. The elementary treatment given here is sufficient for the results of this chapter; the general theory of abstract reduction systems that this construction is an instance of, including the full termination and confluence arguments, is developed independently in Appendix B.

Definition 7.12 (Primitive word)**.** A *primitive word* is a finite sequence $w = \alpha_1\alpha_2\cdots\alpha_n$ with $\alpha_i \in \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$. The empty word is denoted ε.

The collection of all primitive words forms the free monoid $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^*$ under concatenation, containing every syntactically valid computation, including many that are operationally redundant or semantically equivalent; the purpose of rewriting is to identify these equivalent representations.

Definition 7.13 (Rewrite relation)**.** A *rewrite relation* $\longrightarrow$ is a binary relation on $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^*$ such that $u \longrightarrow v$ whenever the primitive word u may be replaced by the simpler word v without altering its admissible structural action on bounded scope geometries.

Repeated application of rewrite rules produces a reduction sequence $w \longrightarrow w_1 \longrightarrow w_2 \longrightarrow \cdots$; a word is *irreducible* if no rewrite rule applies. The primitive algebra admits four elementary reductions, motivated by operational redundancy rather than arbitrary symbolic simplification: identity elimination, $w\varepsilon \longrightarrow w$ and $\varepsilon w \longrightarrow w$ (R1); consecutive refusal, $\mathcal{R}\mathcal{R} \longrightarrow \mathcal{R}$ (R2), since repeated refusals without an intervening admissible transformation are equivalent to a single refusal; consecutive collapse, $\mathcal{C}_l\mathcal{C}_l \longrightarrow \mathcal{C}_l$ (R3), since further collapse operations after a scope has already been collapsed into its canonical representative produce no additional effect; and empty promotion, $\mathcal{P}\varepsilon \longrightarrow \varepsilon$ (R4), since promotion of an empty computation is redundant. These elementary rules intentionally avoid introducing semantic assumptions regarding application-specific behavior; more specialized rewrite rules may be added in concrete implementations provided they preserve admissibility.

Definition 7.14 (Operational equivalence)**.** Two primitive words $u, v \in \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^*$ are *operationally equivalent*, written $u \equiv v$, if one can be transformed into the other through a finite sequence of rewrite rules and inverse rewrite rules.

Operational equivalence partitions the free monoid into equivalence classes whose members describe identical admissible computations, so the Scope Calculus distinguishes between the syntactic form of a computation and its structural effect.

Lemma 7.15 (Termination of elementary rewriting)**.** *Every finite sequence of elementary rewrite rules terminates after a finite number of reductions.*

Proof. Each elementary rewrite rule strictly decreases either the length of the primitive word or the number of redundant operators it contains. Since every primitive word has finite length, no infinite descending chain exists, so every reduction sequence terminates. □

Termination alone is insufficient: different rewrite orders must also lead to compatible results.

Definition 7.16 (Local confluence)**.** A rewriting system is *locally confluent* if whenever $w \longrightarrow u$ and $w \longrightarrow v$, there exists a primitive word z such that $u \longrightarrow^* z$ and $v \longrightarrow^* z$.

For the elementary rewriting rules above, overlapping rewrite patterns occur only in a finite number of cases, each resolving consistently; the detailed verification appears in Appendix I. Termination and local confluence together motivate the search for unique canonical representatives of operational equivalence classes; rather than proving uniqueness immediately, we postpone the complete argument to Chapter 9, where the Primitive Normal Form Theorem is established after the algebraic and categorical foundations have been completed. For the remainder of this chapter a primitive word is called *reduced* if no elementary rewrite rule applies; every admissible computation considered in later chapters may be represented by one or more primitive words, each reducible to a common reduced representative.

7.5 Algebraic Properties of the Scope Calculus

Having introduced the primitive alphabet and its rewriting system, we now examine the algebraic structure it generates. Unlike many familiar algebraic systems, the Scope Calculus is neither a group nor a ring: primitive operations are generally irreversible, context-dependent, and only partially defined. Nevertheless they exhibit a rich collection of structural regularities sufficient to support canonical reduction, categorical interpretation, and representation independence.

Definition 7.17 (Scope monoid)**.** The pair $(\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^*, \circ)$ consisting of all finite primitive words together with concatenation is called the *Scope Monoid*.

Proposition 7.18. *The Scope Monoid is closed under concatenation, associative, and possesses the identity element ε.*

Proof. Closure follows because concatenation of two finite primitive words produces another finite primitive word; associativity follows from concatenation of finite sequences; and ε acts as both left and right identity. □

This monoid is intentionally syntactic; operational semantics arise only after admissibility constraints and rewriting relations are introduced.

Definition 7.19 (Operational quotient)**.** Let $\equiv$ denote operational equivalence under the rewrite system. The quotient algebra $\mathcal{Q} = \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^* / \equiv$ is the *Scope Quotient Algebra*.

Elements of $\mathcal{Q}$ are equivalence classes of primitive computations rather than individual primitive words, so the quotient algebra captures computational behavior instead of syntactic representation.

Definition 7.20 (Partial composition)**.** Let $u, v \in \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^*$. The composition $v \circ u$ is defined only if execution of u terminates in a scope geometry satisfying the admissibility conditions required for the initial application of v.

Thus the Scope Calculus forms a partial algebra over admissible computations; this partiality is not an inconvenience but one of the defining features of the theory, since invalid compositions correspond precisely to structural violations of context.

Proposition 7.21. *Under the elementary rewrite rules, $\mathcal{R}^2 = \mathcal{R}$ and $\mathcal{C}_l^2 = \mathcal{C}_l$.*

Proof. Both identities follow immediately from rewrite rules (R2) and (R3): repeated refusal introduces no additional structural restriction beyond the first refusal, and once a scope has been collapsed into its canonical representative, further collapse operations leave the resulting geometry unchanged. □

These identities illustrate that certain operations are naturally stabilizing, with repeated application eventually reaching a fixed point.

Proposition 7.22. *The Scope Calculus is not, in general, a group.*

Proof. Suppose $\mathcal{C}_l$ possessed an inverse; then every collapsed computation could be reconstructed uniquely into its complete internal history. However, Collapse intentionally replaces temporary internal structure by a canonical representative, so information regarding intermediate computation is generally lost, and $\mathcal{C}_l$ admits no universal inverse. An identical argument applies to $\mathcal{R}$, whose rejection of inadmissible transitions does not uniquely determine the rejected proposal. Hence the primitive algebra cannot satisfy the group axioms. □

This non-invertibility distinguishes Scope Geometry from reversible computation while aligning naturally with evaluation, abstraction, and proof construction.

Definition 7.23 (Operational fixed point)**.** A bounded scope geometry S is a *fixed point* of a primitive transformation α if $\alpha(\mathrm{S}) = \mathrm{S}$.

Examples include a fully evaluated computation under $\mathcal{C}_l$, a boundary already enforcing every relevant admissibility constraint under $\mathcal{R}$, and an established dependency under repeated $\mathcal{B}_d$ where no additional relations are introduced. Fixed points provide the algebraic foundation for convergence arguments developed in Chapter 9 and later become central to the diagnostic theory of stable semantic trajectories in Chapters 13 and 15.

7.6 Discussion

The algebra introduced in this chapter deliberately occupies a middle ground between abstract algebra and operational semantics. On one hand, the primitive operators generate an algebraic structure supporting composition, quotient construction, and canonical reduction; on the other, the algebra remains closely connected to concrete computational behavior through admissibility constraints and contextual boundaries. This dual character distinguishes the Scope Calculus from traditional rewriting systems: rather than manipulating symbols in isolation, every rewrite reflects a structural transformation of bounded semantic contexts. The following chapter abstracts these operational ideas one level further, examining the category whose objects are bounded scope geometries and whose morphisms are admissible context-preserving transformations; the algebra developed here becomes the operational substrate for the categorical constructions of Chapter 8.

7.7 Chapter Summary

This chapter completed the construction of the Scope Calculus. Beginning with the four primitive operations Pop, Refuse, Bind, and Collapse, we developed their composition laws, established the elementary rewriting system, introduced operational equivalence, and constructed the quotient algebra governing admissible computations, showing that the primitive alphabet generates a partial monoid, identifying idempotent operators, proving the absence of global inverses, and introducing operational fixed points. These results provide the complete algebraic machinery required for the categorical development that follows, in which Chapter 8 elevates these operational constructions into the category $\mathbf{Scope}$.

Status of Results: Chapter 7

Established background. Free monoids, rewriting systems, quotient algebras, partial algebras, and operational semantics [5, 29, 37].

Formal developments. Definition of the Scope Monoid, Scope Quotient Algebra, partial composition, operational equivalence, idempotence, non-invertibility, and operational fixed points.

Empirical hypotheses. None. Chapter 7 develops the algebraic machinery underlying the Scope Calculus; empirical application begins only in Volume III after the categorical and topological foundations have been completed.

Chapter 8

Category-Theoretic Foundations of Scope Geometry

Mathematics is the art of giving the same name to different things.
— Henri Poincaré

Category theory was introduced to mathematics by Eilenberg and Mac Lane as a language for describing mathematical structure independently of any particular realization [14, 37], and during the twentieth century it became one of the central organizing frameworks of modern algebra, topology, geometry, logic, and computer science. The motivation for introducing it into Structural Semantics is considerably more modest. The previous chapters established two complementary viewpoints: bounded scopes as mathematical objects possessing boundaries, interiors, and nested child scopes, and admissible computations as finite compositions of primitive operations governed by the Scope Calculus. These developments naturally suggest a higher level of abstraction, in which we study the transformations between scope geometries rather than reasoning directly about primitive words or individual scope objects, and category theory provides precisely the mathematical language required for this transition.

Throughout this chapter we adopt a principle of *structural minimalism*: only those categorical constructions required by subsequent chapters are introduced. We do not assume that the resulting category is monoidal, cartesian closed, complete, cocomplete, or possesses universal limits; such structures may eventually arise as derived properties, but they are never assumed a priori.

8.1 Motivation

Many mathematical theories begin by studying individual objects before discovering that the important information lies not within isolated objects but in the relationships between them: linear algebra studies vector spaces together with linear maps, topology studies spaces together with continuous maps, and group theory studies groups together with homomorphisms. Likewise, Structural Semantics ultimately studies scope geometries together with admissible context-preserving transformations. This shift of emphasis is essential, since a semantic system rarely remains static — programs execute, proofs evolve, conversations accumulate context, reasoning systems revise hypotheses, and large language models continuously transform latent representations from one layer

to the next — so semantic organization should be understood primarily as a family of transformations rather than as isolated states. The purpose of category theory is therefore not to replace the Scope Calculus, but to provide an abstract language describing the algebra constructed in Chapter 7.

Definition 8.1 (Structural minimalism principle)**.** A categorical construction shall be introduced only if every required axiom is explicitly verified, the construction is necessary for subsequent mathematical development, and no stronger categorical structure is assumed than is required.

This principle ensures that the categorical framework grows naturally from the algebra rather than being imposed externally.

8.2 Objects

The objects of the category are precisely the bounded scope geometries introduced earlier, reinterpreted from the categorical perspective.

Definition 8.2 (Objects of Scope)**.** The objects of Scope are all well-formed bounded scope geometries $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$ satisfying the admissibility conditions of Chapter 6.

No assumptions are made concerning the internal structure of the state space: the interior may consist of symbolic expressions, graphs, neural activations, logical formulas, computational processes, or any other representation satisfying the boundary conditions, so the objects of Scope are intentionally substrate independent.

8.3 Morphisms

The morphisms of the category represent admissible semantic transformations. Operationally these correspond to computations generated by the primitive alphabet; abstractly they are structure-preserving maps between bounded scope geometries.

Definition 8.3 (Scope morphism)**.** Let $\mathrm{S}_1, \mathrm{S}_2 \in \mathrm{Ob}(\mathrm{Scope})$. A morphism $\sigma : \mathrm{S}_1 \to \mathrm{S}_2$ is an admissible transformation preserving the boundary relations of the source scope throughout execution.

This definition is deliberately abstract: a morphism is characterized by what it preserves, not by how it is presented syntactically. The connection to the primitive alphabet of Chapter 7 is then a fact to be proved rather than a feature built into the definition, which keeps the category free to accommodate other presentations of the same transformations later.

Proposition 8.4 (Primitive presentation)**.** *Every scope morphism $\sigma \in \mathrm{Hom}(\mathrm{S}_1, \mathrm{S}_2)$ admits a presentation as a finite sequence of primitive operations drawn from $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$.*

Proof. By Definition 6.5, the structural difference between S_1 and S_2 is confined to finite modifications of $\mathcal{B}$, $\mathcal{E}$, and $\mathcal{S}_{\text{child}}$. Theorem 7.6 established that every admissible transformation of a bounded scope geometry decomposes into a finite composition of $\mathcal{P}$, $\mathcal{R}$, $\mathcal{B}_d$, and $\mathcal{C}_l$; applying this to each of the finitely many modifications yields a finite primitive presentation of σ. □

Morphisms need not preserve every internal detail: temporary computational states may change dramatically, and only admissibility constraints are required to remain valid throughout the transformation. This observation foreshadows the Representation Invariance Theorem of Chapter 9. The categorical structure introduced in this chapter is kept deliberately minimal; the stronger question of whether Scope supports the full Cartesian closed structure needed to interpret typed λ-terms is addressed, under explicit hypotheses, in Appendix C.

Definition 8.5 (Identity morphism)**.** For every bounded scope geometry S there exists an identity morphism $\mathrm{id}_{\mathrm{S}} : \mathrm{S} \to \mathrm{S}$ defined by $\mathrm{id}_{\mathrm{S}}(x) = x$ for every admissible state $x \in \mathcal{E}$.

Operationally, the identity morphism corresponds to the empty primitive computation of Chapter 7: no boundaries change, no scope relationships are modified, and no computational history is generated. Identity therefore represents structural persistence rather than computational inactivity.

Definition 8.6 (Composition)**.** Suppose $\sigma_1 : \mathrm{S}_1 \to \mathrm{S}_2$ and $\sigma_2 : \mathrm{S}_2 \to \mathrm{S}_3$. Their composition $\sigma_2 \circ \sigma_1 : \mathrm{S}_1 \to \mathrm{S}_3$ is defined whenever the codomain of σ_1 equals the domain of σ_2.

This definition mirrors ordinary functional composition; operationally it represents the execution of one admissible context transformation immediately after another while preserving the boundary constraints inherited from intermediate scopes.

Theorem 8.7. *The pair* $(\mathrm{Ob}(\mathrm{Scope}), \mathrm{Hom}(\mathrm{Scope}))$ *forms a category.*

Proof. Identity morphisms exist by construction, and composition is defined whenever codomain and domain agree. Associativity follows immediately from associativity of sequential composition established for the Scope Calculus in Chapter 7. Finally, $\mathrm{id}_{\mathrm{S}_2} \circ \sigma = \sigma = \sigma \circ \mathrm{id}_{\mathrm{S}_1}$ for every admissible morphism $\sigma : \mathrm{S}_1 \to \mathrm{S}_2$. Therefore all category axioms are satisfied. □

The importance of this theorem is conceptual rather than computational: it establishes that semantic transformations possess exactly the structural regularity required for categorical reasoning.

8.4 Interpretation

The category Scope should not be interpreted as describing cognition, consciousness, or intelligence; instead it provides an abstract mathematical language for discussing families of admissible transformations independently of their concrete implementation. Whether a computation is carried out by symbolic rewriting, biological neurons, silicon processors, or distributed agents is irrelevant at the categorical level; only the preservation of admissible scope structure remains visible. This abstraction is precisely what later permits comparison between Whiteheadian process philosophy and Scope Geometry through a translation functor developed in Chapter 10. Readers wondering whether the single notion of composition used throughout this chapter is the only one Scope Geometry supports may wish to know that it is not: a second, independent notion of composition, arising from refining a scope rather than executing it, is developed as a conditional extension in Appendix D, though nothing in the chapters that follow depends on it.

8.5 Chapter Summary

This chapter introduced the category Scope using the principle of structural minimalism. Objects were defined as bounded scope geometries, morphisms as admissible context-preserving transformations, and sequential execution was formalized through categorical composition. Verification of the category axioms established that Scope Geometry possesses a mathematically well-defined categorical structure while avoiding unnecessary assumptions concerning monoidal or higher categorical properties. The next chapter develops the principal structural theorems governing this category: closure, decomposition, canonical normal forms, admissibility preservation, reconstruction, and the Representation Invariance Theorem.

Status of Results: Chapter 8

Established background. Elementary category theory: objects, morphisms, identity morphisms, and composition [4, 14, 37].

Formal developments. Definition of the category Scope, the Structural Minimalism Principle, categorical interpretation of bounded scope geometries, admissible morphisms, and verification of the category axioms.

Empirical hypotheses. None. This chapter develops only the abstract mathematical framework required for subsequent structural theorems.

Chapter 9

Foundational Theorems of Structural Semantics

The essence of mathematics lies in its freedom.
— Georg Cantor

The preceding chapters introduced the fundamental components of the Scope Calculus: bounded scope geometries as the primary semantic objects, primitive operations as generators of admissible computations, and Chapter 8 demonstrated that these objects and transformations naturally form the category $\mathbf{Scope}$. The present chapter instead establishes the principal structural results on which the remainder of the monograph depends, providing the internal consistency of the Scope Calculus and justifying its interpretation as a framework for describing semantic organization independently of any particular computational substrate. Several proofs are necessarily lengthy; to preserve continuity of the main exposition, complete technical proofs are deferred to Appendix I, and only the principal arguments are presented here.

9.1 Motivation

Every mathematical theory requires internal guarantees: linear algebra requires that matrix multiplication be associative, topology requires continuity to behave well under composition, and group theory requires closure under multiplication. Structural Semantics requires analogous guarantees, without which the primitive operations of Chapter 7 would remain merely descriptive rather than forming a coherent mathematical system. The goals of this chapter are therefore to demonstrate that admissible computations remain within the space of admissible scope geometries; that every admissible computation possesses a canonical decomposition; that the rewriting system converges to unique normal forms; that semantic preservation can be characterized in terms of admissibility; and finally to establish the Representation Invariance Theorem, which provides the mathematical basis for substrate independence.

9.2 Preliminary Assumptions

Throughout this chapter we assume the framework established previously: bounded scope geometries as in Definition 6.5, primitive operations satisfying the operational

rules of Chapter 7, admissible morphisms forming the category Scope of Chapter 8, and finiteness of every primitive computation. The finiteness assumption is important: without it, uniqueness of normal forms and termination of rewriting would generally require additional hypotheses concerning infinitary rewriting systems.

9.3 Closure of Scope Geometry

Theorem 9.1 (Closure theorem)**.** *Let* $\mathrm{S} \in \mathrm{Ob}(\mathrm{Scope})$ *and let* $\sigma : \mathrm{S} \to \mathrm{S}'$ *be an admissible morphism. Then* $\mathrm{S}' \in \mathrm{Ob}(\mathrm{Scope})$.

Proof. Every admissible morphism preserves the boundary relations defining the target scope. Because admissibility requires every primitive operation to satisfy the boundary tensor $\mathcal{B}$, the resulting object satisfies the same well-formedness conditions required by Definition 6.5, so the codomain remains a bounded scope geometry. A complete inductive proof appears in Appendix I. □

The Closure Theorem guarantees that semantic computation cannot escape the mathematical universe of discourse, analogous to closure properties throughout algebra.

9.4 Canonical Primitive Decomposition

Theorem 9.2 (Scope decomposition)**.** *Every admissible morphism* $\sigma : \mathrm{S}_1 \to \mathrm{S}_2$ *admits a finite decomposition into primitive operations drawn from* $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$.

Proof. Proceed by structural induction on the length of admissible computations. The base case consists of identity morphisms. The inductive step follows because every admissible transformation changes only one of $\mathcal{B}$, $\mathcal{E}$, or $\mathcal{S}_{\text{child}}$, and Chapter 7 established that each such modification is generated by one of the primitive operations. Hence every admissible computation possesses a primitive decomposition. □

The significance of this theorem extends beyond implementation: it states that the four primitive operations are operationally complete with respect to admissible semantic transformations.

9.5 Canonical Normal Forms

Primitive decompositions are generally not unique, since different computational histories may represent the same semantic transformation. The next theorem resolves this ambiguity.

Theorem 9.3 (Primitive normal form)**.** *Every admissible primitive sequence reduces, through the rewriting rules of Chapter 7, to a unique canonical normal form* σ^*.

Proof. Termination follows because every rewrite decreases a well-founded complexity measure on primitive expressions. Local confluence follows from pairwise verification of overlapping rewrite rules. Since the rewriting system is terminating and locally confluent, Newman's Lemma implies global confluence, so every admissible computation possesses a unique normal form. The complete proof occupies Appendix I. □

Without canonical normal forms, semantic comparison between different computations would remain fundamentally ambiguous; normal forms therefore provide the canonical representatives of semantic equivalence classes.

9.6 Admissibility Preservation

Theorem 9.4 (Admissibility preservation)**.** *A computational system possesses structural semantic capacity precisely when every intermediate transformation preserves admissibility.*

Proof. Suppose admissibility fails at some intermediate computation; then at least one boundary relation becomes invalid, and the resulting computation no longer represents a legal transformation within Scope. Conversely, if every intermediate transformation preserves admissibility, every composed morphism remains well-defined by the Closure Theorem, so semantic organization is preserved throughout the computation. □

This theorem formalizes one of the central philosophical claims of Chapter 1: semantic capacity is characterized by preservation of admissible structure rather than by phenomenological or behavioral criteria.

Theorem 9.5 (Composition integrity)**.** *Suppose* $\sigma_1 : \mathrm{S}_1 \to \mathrm{S}_2$ *and* $\sigma_2 : \mathrm{S}_2 \to \mathrm{S}_3$ *are admissible. Then* $\sigma_2 \circ \sigma_1$ *is likewise admissible.*

Proof. Because both morphisms individually preserve the boundary tensor $\mathcal{B}$, every intermediate scope remains admissible. Associativity of composition established in Chapter 8 then implies that their sequential execution also preserves admissibility. □

This theorem allows arbitrarily long semantic computations to be constructed from locally admissible components, providing the compositional principle underlying subsequent chapters.

Theorem 9.6 (State reconstruction)**.** *Let* S *be a bounded scope geometry satisfying the admissibility conditions of Chapter 6. If every collapse operation records its boundary history, then the parent scope can be reconstructed uniquely.*

Proof. Each collapse records precisely those admissible boundary modifications required to reconstruct the parent scope. Because admissibility excludes ambiguous boundary transitions, the reconstruction algorithm is deterministic. Full details are presented in Appendix I. □

This theorem plays an important role later in the development of diagnostic algorithms for transformer models (Chapter 14), where observable trajectories are used to infer latent scope evolution. The underlying pattern — assembling a global object from a family of locally consistent pieces — recurs throughout mathematics under the name of gluing; Appendix F develops this reading of reconstruction explicitly, in terms general enough to state precisely when local reconstruction data fails to determine a unique global scope.

9.7 Representation Invariance

The preceding theorems remain entirely internal: they establish that the calculus is mathematically coherent, but do not yet address its principal philosophical motivation. Structural Semantics was introduced in Chapter 1 as a substrate-independent account of semantic organization. To justify this claim mathematically, we require a precise criterion under which two computational systems possessing different internal architectures may nevertheless instantiate the same semantic computation. Unlike many discussions

of representation in philosophy of mind, the theorem below makes no claims concerning consciousness, subjective experience, intentionality, or ontology; it concerns only the preservation of admissible transformation structure within Scope.

Definition 9.7 (Admissibility-preserving projection)**.** Let $\mathcal{D}$ be a computational state space. A mapping $h : \mathcal{D} \to \mathrm{Ob}(\mathrm{Scope})$ is an *admissibility-preserving projection* if it is surjective; for every admissible transition $x \rightsquigarrow y$ in $\mathcal{D}$, the induced scope transition $h(x) \rightsquigarrow h(y)$ is admissible in Scope; and boundary relations are preserved, $\mathcal{B}(x) = \mathcal{B}(y) \Rightarrow \mathcal{B}(h(x)) = \mathcal{B}(h(y))$.

The definition deliberately does not require injectivity: many computational states may project onto the same scope geometry, since different implementations frequently possess redundant internal degrees of freedom that play no role in semantic organization. The projection should not be interpreted as a decoder recovering hidden semantic content, but as an abstraction analogous to quotient mappings in algebra or projection operators in topology, identifying the structural information retained by the Scope Calculus while ignoring implementation-specific variation.

Remark 9.8. The admissibility clause of Definition 9.7 is a compatibility condition, not a free choice: it requires $x \sim_{\mathrm{adm}} y$ in $\mathcal{D}$ to imply $h(x) \sim_{\mathrm{adm}} h(y)$ in Scope for every pair of states connected by an admissible transition, not merely for some convenient subset of them. Theorem 9.10 below depends on this condition holding for the specific h_1, h_2 under consideration; the theorem is conditional on the existence of such projections; it does not assert that admissibility-preserving projections exist for every pair of systems one might wish to compare.

Definition 9.9 (Semantic equivalence)**.** Let σ_1, σ_2 be admissible morphisms in Scope. They are *semantically equivalent*, $\sigma_1 \sim_{\mathrm{sem}} \sigma_2$, whenever their canonical normal forms coincide: $\sigma_1^* = \sigma_2^*$.

Because canonical normal forms are unique (Theorem 9.3), this relation is well defined, and it is straightforward to check that $\sim_{\mathrm{sem}}$ is an equivalence relation on the class of admissible morphisms; the admissible morphisms therefore partition into equivalence classes, each represented uniquely by its canonical normal form.

Theorem 9.10 (Representation invariance)**.** *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be computational systems with state spaces $\mathcal{D}_1, \mathcal{D}_2$. Suppose there exist admissibility-preserving projections $h_1 : \mathcal{D}_1 \to \mathrm{Ob}(\mathrm{Scope})$ and $h_2 : \mathcal{D}_2 \to \mathrm{Ob}(\mathrm{Scope})$ such that every admissible computation induces a morphism of Scope; corresponding induced morphisms possess identical canonical normal forms; and corresponding boundary tensors are preserved under h_1 and h_2. Then $\mathcal{M}_1$ and $\mathcal{M}_2$ are semantically equivalent relative to the Scope Calculus.*

Proof. Let $T_1 = (x_0, \dots, x_n)$ be an admissible computation inside $\mathcal{M}_1$, and let $T_2 = (y_0, \dots, y_m)$ be the corresponding computation inside $\mathcal{M}_2$. Applying the projections yields admissible morphisms $\sigma_1 = h_1(T_1)$ and $\sigma_2 = h_2(T_2)$. By Definition 9.7, both morphisms preserve admissibility. By Theorem 9.3, each possesses a unique canonical normal form, σ_1^*, σ_2^*. The second hypothesis states $\sigma_1^* = \sigma_2^*$, hence $\sigma_1 \sim_{\mathrm{sem}} \sigma_2$ by Definition 9.9. Preservation of boundary tensors guarantees that the equivalence extends beyond the terminal states to every admissible intermediate transition. Therefore both systems determine precisely the same semantic computation inside Scope, and $\mathcal{M}_1, \mathcal{M}_2$ are semantically equivalent relative to the Scope Calculus. □

The theorem deliberately avoids stronger claims: it does not assert that two computational systems are physically equivalent, nor that they have equal computational complexity, energy consumption, learning dynamics, or phenomenological equivalence. It establishes only that the admissible semantic structure recognized by the Scope Calculus is invariant under the stated projection hypotheses.

Corollary 9.11. *Structural semantic capacity depends only on admissible transformation structure and not on the physical substrate implementing that structure.*

Proof. Immediate from Theorem 9.10. □

The Representation Invariance Theorem serves as the principal mathematical bridge between the abstract theory of Volumes I and II and the empirical investigations of transformer representations developed in Volume III: rather than searching for semantic organization directly within vectors, tensors, symbolic graphs, or biological neurons, subsequent chapters investigate whether these systems induce equivalent admissible transformation structures after projection into the common categorical framework established by Scope.

9.7.1 Interpretation of Representation Invariance

The theorem should not be understood as claiming that all computational architectures are equivalent, but as specifying a precise criterion under which equivalence may be established — one based not on physical realization, implementation language, computational complexity, or biological plausibility, but solely on the preservation of admissible transformation structure within Scope. This mirrors analogous developments throughout mathematics: the same abstract group may be represented by matrices, permutations, or symmetries of geometric objects, and although these realizations differ substantially in implementation, they are regarded as equivalent because they preserve the same algebraic structure. Structural Semantics adopts an analogous perspective, treating the organization of admissible transformations rather than the implementation or coordinate system as the object of interest.

Definition 9.12 (Substrate independence)**.** A semantic property is *substrate independent* whenever its validity depends only on admissible structural transformations within Scope and not on the physical realization of those transformations.

This definition intentionally avoids stronger metaphysical claims: it does not imply that all substrates are equally efficient, expressive, or capable of supporting arbitrary computations, only that once admissible transformations have been projected into the Scope Calculus, semantic evaluation depends on those transformations rather than on their physical implementation. Consider, for instance, a compiler translating a high-level programming language into two different machine architectures: although the resulting executable instructions differ substantially, both preserve identical control flow, scope hierarchy, and variable-binding relationships, and within the Scope Calculus these executions correspond to the same canonical admissible morphism despite differing binary representations. Similarly, if a symbolic reasoning system built on graph rewriting and a transformer neural network induce identical admissible primitive transformations after projection into Scope, Theorem 9.10 establishes their semantic equivalence relative to the Scope Calculus, without implying identical learning mechanisms, computational cost, training dynamics, or phenomenological properties.

Representation Invariance occupies an intermediate position between several established traditions: denotational equivalence in formal semantics identifies programs whose observable behaviors coincide despite differing implementations [49, 58], quotient constructions in algebra identify objects modulo equivalence relations, and naturally isomorphic constructions in category theory are routinely regarded as equivalent despite differing internal presentations [4, 37]. The present theorem shares this structural philosophy while preserving a different invariant: instead of numerical values, logical formulas, or observable program outputs, it preserves admissible context transformations. This distinction becomes particularly important in Chapters 13–15, where the empirical question investigated is not whether latent vectors resemble human concepts directly, but whether they preserve admissible semantic organization under projection into Scope Geometry.

Remark 9.13 (A parallel in philosophy of physics). A structurally similar move appears in the philosophy of physics. Ismael and van Fraassen [32] argue that a symmetry transformation identifies surplus theoretical structure precisely when it leaves every observable prediction unchanged: two models related by such a transformation, $M \sim M'$, should be treated as one physical state rather than two, so that the physically meaningful object is the quotient $\mathcal{M}/\sim$ rather than the original space of models. The admissibility-preserving projections of Definition 9.7 play an analogous role here: two computational systems related by such a projection are treated, relative to the Scope Calculus, as one semantic computation rather than two, regardless of how their underlying representations differ. The two constructions are not the same — theirs identifies structure that is dynamically inert, ours identifies structure that is admissibility-irrelevant — but both replace a space of concrete representations with a coarser space of what those representations are for. Separately, Ismael [31] argues that a deterministic process cannot introduce a symmetry-breaking distinction into its effect that was not already present, in some form, in its cause: an asymmetric outcome requires an asymmetric antecedent. The Closure Theorem (Theorem 9.1) and Admissibility Preservation Theorem (Theorem 9.4) make a comparable claim about admissible transformations specifically: no admissible sequence of primitive operations can produce a scope geometry violating constraints that were not already violable given the boundary structure it started from. Neither parallel is used to derive anything in this monograph; both are noted because the same structural intuition — that transformation can reveal or reorganize distinctions but not manufacture them from nothing — recurs across fields that otherwise share little vocabulary.

9.7.2 Representations as Coordinates Rather Than Meaning

It is worth stating plainly what Theorem 9.10 implies about the status of representations themselves, since the implication is easy to state informally but easy to misread formally. The theorem does not say that vectors, symbols, neural activations, and graphs are all secretly the same kind of thing. It says that, once an admissibility-preserving projection into Scope has been specified, the differences among them become differences in *coordinates* on a common invariant object rather than differences in the object itself — in the same sense that a point in the plane has different numerical descriptions in Cartesian and polar coordinates without being two different points.

This reframing has a consequence for how the question "what does this representation mean?" should be posed within this framework. Asked of a coordinate system directly, the question is comparatively hard: a 4096-dimensional activation vector, consid-

ered as raw numbers, carries no more intrinsic meaning than a pair of polar coordinates carries intrinsic direction independent of the convention that assigns it one. Asked of the admissible transformation the vector participates in, the question becomes tractable, because admissibility is exactly the structure the projection is built to preserve. The shift in emphasis from Chapters 11 through 13, from cataloguing what activation space *looks like* to tracking what transformations it *admits*, is this reframing applied to a specific representational format.

The reframing has a limit worth stating alongside it. A choice of coordinates is never entirely arbitrary in practice: some coordinate systems make a given computation efficient and others make it intractable, exactly as Cartesian coordinates simplify some geometric problems that polar coordinates complicate. Representation Invariance establishes that semantic content, as this monograph defines it, does not depend on the choice; it does not establish that the choice is inconsequential for anything else, including learnability, computational cost, or the ease with which a projection satisfying Definition 9.7 can be found or verified in the first place.

9.7.3 Limitations of the Present Results

The Representation Invariance Theorem establishes only a conditional equivalence: its conclusions depend on the hypotheses stated explicitly in Theorem 9.10, and if admissibility-preserving projections do not exist, or if canonical normal forms fail to coincide, no semantic equivalence follows. Nor does the theorem provide a constructive algorithm for discovering such projections; the existence of admissibility-preserving mappings is a mathematical assumption whose verification depends on the computational architecture under investigation, and developing practical procedures for constructing these projections is one of the principal objectives of Volume III. The distinction between existential theorems and constructive algorithms is fundamental: Chapter 9 proves that semantic equivalence follows if suitable projections exist, while Chapter 14 investigates algorithmic procedures for approximating such projections within transformer activation spaces. The mathematical theorem should therefore not be interpreted as an implemented diagnostic method, but as the theoretical target toward which the diagnostic algorithms are directed.

9.8 Levels of Structural Equivalence

The equivalence established by Theorem 9.10 is exact: two systems are semantically equivalent relative to the Scope Calculus only if their projected morphisms reduce to the identical canonical normal form. Part III will repeatedly ask a looser question — whether two activation trajectories are similar enough, in some measurable sense, to indicate comparable semantic organization — and it is worth distinguishing, before that material begins, several notions of equivalence that exact identity does not exhaust. Conflating them is a natural source of overclaiming in exactly the kind of empirical work Chapters 13 through 15 undertake.

Exact equivalence is the relation of Definition 9.9: two morphisms are equivalent only if $\sigma_1^* = \sigma_2^*$, with no tolerance. This is the only notion of equivalence any theorem in this monograph establishes.

Approximate equivalence weakens exact equivalence by a distance threshold: two morphisms might be considered equivalent if their canonical normal forms differ by less than some ϵ under a suitable metric on $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^*$. Nothing in Chapters 6–9

constructs such a metric or proves that any candidate metric interacts well with composition, so approximate equivalence in this sense remains an open extension rather than an established consequence of the present theorems.

Probabilistic equivalence replaces a fixed morphism with a distribution over morphisms, as arises naturally once retrieval or routing in a real architecture is stochastic rather than deterministic; two systems might then be called equivalent if their induced distributions over canonical normal forms coincide or are close in some divergence. Appendix G's measure-theoretic constructions provide some of the vocabulary such a notion would need, but the monograph does not develop it into a theorem.

Functional equivalence is weaker still, and closest to what Chapters 14 and 15 can actually test: two systems are functionally equivalent if they produce statistically indistinguishable diagnostic behavior — comparable Phase Locking Values, comparable admissibility estimates — without any claim that their underlying admissible transformations coincide, approximately or otherwise. This is the notion implicitly in play whenever an experiment in Chapter 15 compares model architectures rather than proving a theorem about them.

The four levels form a strict hierarchy: exact equivalence implies every weaker notion, but none of the implications reverse. An empirical finding of functional equivalence between two systems is evidence bearing on, but not a proof of, probabilistic, approximate, or exact equivalence in the senses above. Keeping this hierarchy in view is part of what the Methodological Statement of the front matter asks of every claim in this monograph: an empirical result belongs to the empirical-hypothesis category regardless of how strongly it is worded, and dressing a functional-equivalence finding in the language of Theorem 9.10 would misstate what has actually been shown.

9.9 Chapter Summary

This chapter established the principal mathematical foundations of Structural Semantics. Beginning with the Closure Theorem, we demonstrated that admissible computations remain within the category Scope. The Scope Decomposition and Primitive Normal Form Theorems showed that every admissible computation possesses a canonical structural representation. The Admissibility Preservation and Composition Integrity Theorems characterized semantic organization as the preservation of boundary constraints under sequential transformation, while the State Reconstruction Theorem established that admissible computational histories can be recovered from recorded boundary events. These internal consistency results culminated in the Representation Invariance Theorem, which demonstrated that semantic equivalence depends on admissible transformation structure rather than the underlying computational substrate, providing the mathematical foundation for treating symbolic systems, neural architectures, graph-based reasoners, and other computational models within a common structural framework. The next chapter builds on these results by introducing a translation functor from Whiteheadian process ontology into Scope Geometry.

Status of Results: Chapter 9

Established background. Canonical rewriting systems, equivalence relations, quotient constructions, denotational semantics, and elementary category theory [4, 37, 49, 58].

Formal developments. The Closure, Scope Decomposition, Primitive Normal Form, Admissibility Preservation, Composition Integrity, and State Reconstruction Theorems, together with the Representation Invariance Theorem and its corollary establishing substrate-independent structural semantic capacity.

Empirical hypotheses. None. All results established in this chapter are purely mathematical, conditional on the stated hypotheses. The construction of practical admissibility-preserving projections is deferred to the computational framework developed in Volume III.

Chapter 10

Comparative Ontology and the Translation Functor

The many become one, and are increased by one.
— Alfred North Whitehead, *Process and Reality*

The preceding chapter established the internal mathematical foundations of Structural Semantics: closure, canonical decomposition, admissibility preservation, and representation invariance demonstrated that the Scope Calculus forms a coherent mathematical system independent of any particular computational substrate. The present chapter serves a different purpose. Rather than proving additional properties of the calculus itself, we construct a formal correspondence between the Scope Calculus and an existing philosophical framework, asking whether the structural relationships described by Whitehead's process ontology admit a mathematically precise translation into the language of bounded scope geometries. The answer developed here is affirmative, although carefully qualified: we define a functor $F : \mathrm{Process} \to \mathrm{Scope}$ that preserves the operational structure of process interactions while explicitly identifying the point at which the two theories diverge, namely Whitehead's notion of Subjective Form.

10.1 Motivation

A recurring theme of twentieth-century philosophy is that process precedes substance: rather than viewing reality as composed of static objects possessing accidental properties, process philosophies understand stable objects as relatively persistent organizations of ongoing events. Although Whitehead developed this view for metaphysical purposes, remarkably similar ideas appear throughout modern theoretical computer science — programming languages describe evolving computational states, distributed systems describe interacting processes, category theory emphasizes morphisms before objects, dynamical systems focus on trajectories rather than isolated points, and large language models themselves perform successive transformations of latent representations rather than manipulating static symbolic objects. These parallels motivate a careful mathematical comparison. The purpose of this chapter is therefore not to defend Whiteheadian metaphysics, nor to derive the Scope Calculus from process philosophy, but to investigate the extent to which the operational structure of one framework may be translated into the other.

10.2 The Category of Processes

Throughout this chapter we use $\mathrm{Process}$ to denote an abstract category whose objects represent Whiteheadian actual occasions and whose morphisms represent admissible process transitions. This construction is not a complete formalization of Whitehead's philosophy, but a mathematical abstraction capturing only those structures required for comparison with Scope Geometry.

Definition 10.1 (Category of processes)**.** The category $\mathrm{Process}$ consists of objects representing actual occasions, morphisms representing admissible transitions between occasions, ordinary categorical composition, and identity morphisms corresponding to persistence of an occasion.

This definition intentionally avoids introducing unnecessary metaphysical structure; concepts such as eternal objects, creativity, and the extensive continuum remain outside the scope of the present construction.

Definition 10.2 (Actual occasion, categorical form)**.** An actual occasion is represented by the ordered triple $P = \langle D, v, A \rangle$, where D denotes the inherited data, v the polarity of prehension, and A the Subjective Form associated with the occasion.

This representation follows the operational interpretation of Chapter 4 and serves as the domain of the translation functor; no assumptions are yet made concerning the interpretation of A, whose treatment emerges naturally from the construction of the functor.

10.3 Operational Correspondence

Inspection of Chapters 4 through 9 reveals a striking operational parallel between the two theories: positive prehensions accumulate structural relations, negative prehensions exclude incompatible relations, transduction propagates contextual information, and concrescence produces completed organizations, while the Scope Calculus possesses analogous operations in binding, refusal, promotion, and canonical evaluation. This observation motivates the correspondence summarized in Table 10.1.

Whiteheadian process	Scope Calculus
Positive prehension	$\mathcal{B}_d$
Negative prehension	$\mathcal{R}$
Transduction	$\mathcal{P}$
Concrescence	$\mathcal{C}_l$

Table 10.1: Operational correspondence between process ontology and Scope Geometry.

The purpose of the remainder of the chapter is to determine whether this informal correspondence extends to a genuine functor.

Definition 10.3 (Translation functor)**.** The translation functor $F : \mathrm{Process} \to \mathrm{Scope}$ acts on objects according to $F(P) = \mathrm{S}$, where $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$ is the corresponding bounded scope geometry, and on morphisms maps process transitions to admissible scope morphisms preserving boundary structure.

The operational component of the translation is determined by the assignments $F(P^+) = \mathcal{B}_d$, $F(P^-) = \mathcal{R}$, $F(T) = \mathcal{P}$, and $F(C) = \mathcal{C}_l$, motivated by operational rather than philosophical similarity: each pair performs analogous structural functions despite arising from different theoretical traditions, and the interpretation should be understood as structural rather than metaphysical. The remaining question concerns Subjective Form, which unlike the preceding components has no direct analogue inside the Scope Calculus and requires separate consideration.

Example 10.4 (The functor applied to a single occasion)**.** Let $P = \langle D, v, A\rangle$ describe an occasion prehending two antecedent data, d_1 positively and d_2 negatively. Under F, the positive prehension of d_1 becomes an instance of $\mathcal{B}_d$, establishing an admissible dependency between d_1 and the developing occasion within the corresponding scope; the negative prehension of d_2 becomes an instance of $\mathcal{R}$, excluding d_2 from the interior of that scope. If the occasion's data derive from an antecedent occasion in a different scope, the transduction carrying that data forward becomes an instance of $\mathcal{P}$; the occasion's concrescence into a completed satisfaction becomes an instance of $\mathcal{C}_l$, the scope's resolution into a canonical result. Every component of P except A is accounted for by one of these four assignments; nothing in the translation invents a fifth operator or maps prehension onto a notion — such as a labeled "inclusion vector" or "exclusion predicate" — outside the primitive alphabet $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$ already fixed in Chapter 7.

10.4 Functorial Preservation

Having defined the action of the translation functor on both objects and primitive operations, we must verify that it satisfies the defining axioms of a functor: preservation of identity morphisms and of composition. Recall from Chapter 8 that a functor between two categories preserves the categorical structure, so the validity of the present construction depends on ordinary categorical reasoning [4, 37] rather than philosophical interpretation. This is the second functor constructed in this monograph rather than the first: Appendix C constructs one between the simply typed λ-calculus and Scope under a Cartesian closure hypothesis, and the categorical vocabulary used to verify F here is the same vocabulary developed rigorously there.

Proposition 10.5 (Identity preservation)**.** *For every object $P \in \mathrm{Process}$, the translation functor satisfies $F(\mathrm{id}_P) = \mathrm{id}_{F(P)}$.*

Proof. Let $P = P$ be an arbitrary process object. The identity morphism id_P produces no modification of inherited data, process polarity, or admissible transition structure. Under the action of the translation functor, $P \mapsto \mathrm{S}$; since no structural modification occurs, the corresponding scope geometry also remains unchanged, so $F(\mathrm{id}_P) = \mathrm{id}_{\mathrm{S}} = \mathrm{id}_{F(P)}$. □

Theorem 10.6 (Composition preservation)**.** *Let $f : P_1 \to P_2$ and $g : P_2 \to P_3$ be morphisms in* Process*. Then $F(g \circ f) = F(g) \circ F(f)$.*

Proof. Suppose f induces the primitive sequence $(\alpha_1, \ldots, \alpha_n)$ while g induces $(\beta_1, \ldots, \beta_m)$. Composition in $\mathrm{Process}$ corresponds to concatenation of these transition sequences. The translation functor maps each primitive process operation to its corresponding primitive scope operation according to the assignments above, so $F(g \circ f)$ produces exactly the concatenated admissible primitive sequence $F(\beta_1), \ldots, F(\beta_m), F(\alpha_1), \ldots, F(\alpha_n)$, identical to first applying $F(f)$ followed by $F(g)$. Hence $F(g \circ f) = F(g) \circ F(f)$. □

10.5 The Image of Subjective Form

The previous sections established a nearly complete structural correspondence between process ontology and Scope Geometry. One component, Subjective Form (A), has remained untouched, and this absence is not accidental: it marks the precise mathematical location at which the two frameworks diverge.

Definition 10.7 (Image of the functor)**.** The image of $F : \mathrm{Process} \to \mathrm{Scope}$ consists of every object and morphism generated solely through admissible scope operations.

Theorem 10.8 (Phenomenological boundary)**.** *Subjective Form A lies outside the image of the translation functor: $A \notin \mathrm{Im}(F)$.*

Proof. The Scope Calculus is generated entirely by the primitive operational alphabet $\{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$. Every object and morphism in Scope is constructed by finite compositions of these operations together with their induced boundary relations. Subjective Form A does not participate in any primitive operational rule: it neither modifies admissibility conditions nor induces additional scope transformations, so no admissible sequence of primitive operations generates an object corresponding to it. Therefore A does not belong to the image of the functor. □

Theorem 10.8 should not be interpreted as a refutation of Whitehead's philosophy, but as identifying the precise boundary between two theories: the translation functor preserves every operational component of process organization represented within the Scope Calculus, while Subjective Form remains external because it is introduced as an additional primitive of the philosophical theory rather than as an operational component of the structural calculus. This distinction mirrors the methodological boundary established in Chapter 5 between formal mathematical constructions and phenomenological interpretations.

10.6 Interpretive Consequences

The existence of the translation functor demonstrates that substantial portions of Whitehead's process ontology admit an entirely structural interpretation: positive and negative prehensions become admissible binding and refusal operations, concrescence becomes canonical collapse, and propagation of process becomes scope transduction. The resulting correspondence is remarkably extensive, but not complete, and its incompleteness is mathematically informative rather than philosophically problematic. Instead of attempting to eliminate Subjective Form or reinterpret it as a hidden computational variable, the present framework identifies the limits of the translation: the image of the functor consists precisely of those aspects of process ontology that admit operational realization within the Scope Calculus. Structural Semantics therefore neither affirms nor denies the existence of phenomenological experience; it demonstrates only that the structural organization of semantic transformations can be developed without requiring phenomenological primitives, while whether additional phenomenological structure exists remains an independent philosophical question lying outside the mathematical scope of the present theory.

10.7 Chapter Summary

This chapter constructed the translation functor $F : \mathrm{Process} \to \mathrm{Scope}$, verified that it preserves identity morphisms and composition, and established that Subjective Form lies outside its image, giving a formal mathematical expression of the phenomenological boundary established in Chapter 5. The next chapter leaves philosophy behind and turns toward contemporary machine learning, examining the mathematical structure of transformer architectures before introducing the novel dynamical and spectral analyses developed later in Volume III.

Status of Results: Chapter 10

Established background. Elementary category theory, functors, and Whitehead's process ontology as reconstructed in Chapters 4–5 [4, 37, 56].

Formal developments. The category $\mathrm{Process}$, the translation functor $F : \mathrm{Process} \to \mathrm{Scope}$, proofs of identity and composition preservation, and the Phenomenological Boundary Theorem establishing that Subjective Form lies outside the image of F.

Empirical hypotheses. None. This chapter is a purely mathematical comparative construction.

Part III

Computational Semantics: Transformer Diagnostics

Chapter 11

The Mathematical Structure of Transformer Architectures

Attention is all you need.
— Ashish Vaswani et al., 2017

The previous volume developed the abstract mathematical framework of Structural Semantics independently of any particular computational substrate: Scope Geometry, admissibility, categorical structure, and the Representation Invariance Theorem were formulated without reference to neural networks, symbolic systems, or biological cognition. Having established this substrate-independent foundation, we now turn to one important computational realization, modern transformer architectures. This chapter differs fundamentally from those preceding it: no new mathematical framework is introduced, and no novel definitions of Structural Semantics are proposed. It instead serves as a survey of established mathematical results concerning transformer architectures, establishing a common notation and mathematical vocabulary for the remainder of Volume III. Maintaining this separation matters methodologically: the constructions introduced later, most notably the activation trajectory framework of Chapter 12 and the resonant analytic signal model of Chapter 13, are intended as original proposals that should be evaluated against an accurate presentation of the current state of transformer theory rather than replacing or reinterpreting it.

11.1 Historical Development

The transformer architecture was introduced by Vaswani et al. [55] as an alternative to recurrent and convolutional architectures for sequence modeling. Rather than processing sequences strictly in temporal order, transformers employ attention mechanisms that permit every token to interact directly with every other token within the available context window. This architectural change increased computational parallelism, since token representations could be processed simultaneously rather than recursively; made long-range dependencies easier to model, since every token could attend directly to distant contextual information; and proved highly scalable, with increasing parameter count, model depth, and training corpus size producing remarkably predictable improvements across a wide variety of language tasks [27, 33]. These developments established transformers as the dominant architecture for contemporary large language models.

11.2 Residual Stream Representation

Mechanistic interpretability has converged on the residual stream as one of the principal mathematical objects of transformer analysis [15, 42, 44]. Throughout this monograph we adopt the standard notation $\mathrm{X} \in \mathbb{R}^{(L+1)\times T\times d_{\text{model}}}$, where L denotes the number of transformer layers, T the sequence length, and d_{model} the embedding dimension; each element $\mathrm{x}_{l,t} \in \mathbb{R}^{d_{\text{model}}}$ represents the residual vector corresponding to token t after completion of layer l. The residual stream functions as a persistent information channel through which every attention block and multilayer perceptron contributes incremental modifications, providing a natural mathematical object for analysing evolving semantic organization.

11.3 Attention Mechanisms

Self-attention constitutes the defining computational mechanism of the transformer architecture. Given an input matrix $X \in \mathbb{R}^{T\times d}$, linear projections generate query, key, and value matrices $Q = XW_Q$, $K = XW_K$, $V = XW_V$, and scaled dot-product attention is computed as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V,$$

where d_k denotes the key dimension. Multi-head attention repeats this computation across several independent projection spaces before concatenating the resulting representations. From the perspective of Structural Semantics, attention itself should be regarded simply as an established computational mechanism; no structural interpretation is imposed at this stage, and such interpretations are deferred until the activation trajectory framework of Chapter 12.

11.4 Feed-Forward Networks

Each transformer layer also contains a position-wise multilayer perceptron, typically represented as $\text{MLP}(x) = W_2\,\phi(W_1 x + b_1) + b_2$, where ϕ denotes a nonlinear activation function such as GELU or SwiGLU. Although attention is often viewed as the defining component of transformers, mechanistic interpretability studies increasingly recognize MLP layers as major repositories of learned features [21], so the residual stream reflects the combined effects of attention and feed-forward computation rather than either mechanism individually.

11.5 Activation Manifolds

Each residual vector $\mathrm{x}_{l,t}$ may be regarded as a point within the high-dimensional space $\mathbb{R}^{d_{\text{model}}}$, and as successive transformer layers are evaluated these vectors trace continuous trajectories through activation space. The collection of all such vectors defines a high-dimensional activation manifold whose geometry has become a major topic of mechanistic interpretability research, investigated through cosine similarity, Euclidean distance, principal-component structure, singular-vector decompositions, sparse feature representations, clustering, activation patching, and causal mediation. These methods have substantially improved understanding of transformer representations, but remain

primarily geometric descriptions of isolated layers or local neighborhoods. The following chapter introduces a complementary viewpoint in which activation vectors are interpreted as continuous trajectories evolving through depth rather than as independent static points.

11.6 Current Directions in Mechanistic Interpretability

Recent work in mechanistic interpretability has shifted from identifying individual neurons toward understanding distributed computational circuits, including induction heads, composition of attention circuits, feature superposition, sparse autoencoders, causal tracing, and activation patching [2, 15, 52]. These studies share the common objective of seeking explicit mathematical explanations for the computations performed by internal representations rather than treating neural networks as opaque statistical devices. Structural Semantics aligns naturally with this broader objective, though the framework developed in the present monograph differs in emphasis: whereas mechanistic interpretability often studies the functional role of particular neurons or circuits, subsequent chapters focus on admissibility-preserving trajectories through activation space, shifting the object of study from isolated computational units to the global organization of transformations.

11.7 Relationship to Scope Geometry

At this point no mathematical identification is made between transformer representations and Scope Geometry; this restraint is deliberate. Chapter 9 established that semantic equivalence requires explicit admissibility-preserving projections into Scope, and the existence of such projections cannot simply be assumed for neural activation spaces. Instead the remainder of Volume III develops these constructions gradually: Chapter 12 introduces activation trajectories as continuous dynamical systems, Chapter 13 constructs analytic signal representations of those trajectories, Chapter 14 develops computational algorithms for estimating admissibility within latent representations, and Chapter 15 proposes empirical experiments capable of evaluating these constructions.

11.8 Chapter Summary

This chapter reviewed the established mathematical foundations of transformer architectures, introducing the residual stream tensor, attention mechanisms, feed-forward networks, activation manifolds, and current approaches in mechanistic interpretability while adopting the standard notation used throughout the contemporary literature. Importantly, no novel theoretical claims were introduced; the purpose has been exclusively expository. The next chapter begins the original developments of Volume III by replacing the conventional view of activation vectors as isolated points with a dynamical interpretation in which semantic organization is studied through continuous trajectories evolving across transformer depth.

Status of Results: Chapter 11

Established background. Transformer architecture, self-attention, residual streams, feed-forward networks, activation manifolds, scaling laws, and mechanistic interpretabil-

ity [15, 27, 33, 42, 44, 55].

Formal developments. None. This chapter establishes notation, terminology, and mathematical background only.

Empirical hypotheses. None. Novel empirical hypotheses begin in Chapter 12 with the dynamical analysis of activation trajectories.

Chapter 12

Dynamic Geometry of Activation Trajectories

The mathematics of change is not merely the mathematics of states, but of the paths connecting them.
— inspired by the dynamical systems tradition

The previous chapter introduced the standard mathematical formalism used to describe transformer architectures, establishing a common vocabulary without introducing new theory. This chapter begins the original developments of Volume III. Rather than studying activation vectors as isolated points in a high-dimensional space, we examine the trajectories generated as those vectors evolve throughout the forward computation of a transformer, a shift from static to dynamical geometry paralleling a well-established transition throughout mathematics and physics – differential geometry studies curves rather than isolated coordinates, dynamical systems investigate flows rather than individual states, and control theory analyzes trajectories rather than instantaneous configurations. The constructions developed in this chapter remain entirely within conventional mathematics, employing only linear algebra, multivariable calculus, and classical dynamical systems theory; the analytic signal formulation of Chapter 13 is deliberately deferred until this ordinary dynamical framework has been established.

12.1 Activation Trajectories

Consider a transformer of L layers. For a fixed input token t, the residual stream produces a sequence of vectors $\mathrm{x}_{0,t}, \mathrm{x}_{1,t}, \dots, \mathrm{x}_{L,t}$, with $\mathrm{x}_{l,t} \in \mathbb{R}^{d_{\text{model}}}$. Rather than treating these vectors independently, we interpret them as a trajectory through activation space.

Definition 12.1 (Activation trajectory)**.** For a fixed token t, the activation trajectory is the ordered sequence $\gamma_t = (\mathrm{x}_{0,t}, \mathrm{x}_{1,t}, \dots, \mathrm{x}_{L,t})$; equivalently, $\gamma_t : \{0, \dots, L\} \to \mathbb{R}^{d_{\text{model}}}$ is a discrete curve parameterized by transformer depth.

This trajectory represents the progressive refinement of the latent representation associated with a single token as information propagates through the network. Where Chapter 11 emphasized the geometry of activation space itself, the present chapter emphasizes the geometry of motion within that space.

12.2 Discrete Dynamical Systems Interpretation

Transformer evaluation naturally induces a discrete dynamical system: each layer defines a transition operator $F_l : \mathbb{R}^{d_{\text{model}}} \to \mathbb{R}^{d_{\text{model}}}$ such that $\mathrm{x}_{l+1} = F_l(\mathrm{x}_l)$, and the complete forward computation is the composition $F = F_{L-1} \circ \cdots \circ F_1 \circ F_0$. Unlike autonomous dynamical systems, transformer dynamics are generally layer-dependent, since each layer possesses distinct learned parameters, so $F_l \neq F_k$ for $l \neq k$ and the resulting system is non-autonomous. Nevertheless, many classical concepts from dynamical systems theory remain applicable.

12.3 Layerwise Jacobians and Local Linearization

Definition 12.2 (Layerwise Jacobian)**.** For each layer F_l, the layerwise Jacobian is $J_l = \partial \mathrm{x}_{l+1} / \partial \mathrm{x}_l$.

The Jacobian describes the local sensitivity of subsequent activations to small perturbations of the current activation vector; large singular values correspond to locally expansive directions, small singular values to locally contractive directions, and the eigenspectrum of J_l therefore provides valuable information concerning trajectory stability. Classical dynamical systems theory studies nonlinear trajectories through their local linear approximations: for a sufficiently small perturbation $\delta \mathrm{x}_l$, a first-order Taylor expansion yields $\delta \mathrm{x}_{l+1} = J_l \, \delta \mathrm{x}_l + O(\|\delta \mathrm{x}_l\|^2)$, so that to first order J_l completely determines the local evolution of perturbations and local trajectory stability can be analyzed independently of the global nonlinear dynamics.

12.4 Curvature and Trajectory Length

Activation trajectories rarely follow straight paths through latent space; instead, successive layers continually redirect the representation, motivating the introduction of curvature.

Definition 12.3 (Discrete curvature)**.** Let $\Delta_l = \mathrm{x}_{l+1} - \mathrm{x}_l$. The discrete curvature at layer l is $\kappa_l = \|\Delta_{l+1} - \Delta_l\|$.

Curvature measures the rate at which the direction of the activation trajectory changes between successive layers: large curvature indicates rapid representational restructuring, small curvature relatively stable semantic evolution. Unlike the spectral quantities of the following chapter, curvature depends only on elementary Euclidean geometry.

Definition 12.4 (Trajectory length)**.** The length of an activation trajectory is $\mathcal{L}(\gamma) = \sum_{l=0}^{L-1} \|\mathrm{x}_{l+1} - \mathrm{x}_l\|$.

Trajectory length measures the total representational displacement experienced during forward computation. Different transformer architectures frequently exhibit substantially different trajectory lengths despite achieving similar downstream task performance, consequently providing a useful architecture-independent geometric statistic.

12.5 Local Stability and Lyapunov Analysis

One of the central concepts of dynamical systems theory is the Lyapunov exponent. Although transformers are finite discrete systems rather than continuous flows, an analogous local quantity may still be defined.

Definition 12.5 (Local layerwise Lyapunov exponent)**.** For each layer l, define $\lambda_l = \log \sigma_{\max}(J_l)$, where $\sigma_{\max}(J_l)$ denotes the largest singular value of the Jacobian.

Positive values indicate local expansion, negative values contraction, and values near zero approximately neutral dynamics; these quantities describe only local behavior and should not be confused with global Lyapunov exponents defined for autonomous dynamical systems. The singular value decomposition $J_l = U_l \Sigma_l V_l^\top$ provides an orthogonal decomposition of the local tangent space, in which the right singular vectors describe perturbation directions entering the layer, the left singular vectors the corresponding output directions, and the singular values the local amplification, permitting activation trajectories to be analyzed in terms of dominant expansion and contraction modes. Later chapters reinterpret these modes using analytic signal representations.

12.6 Relationship to Scope Geometry

The mathematical constructions introduced thus far remain entirely within ordinary dynamical systems theory; no assumptions have yet been made regarding admissibility, boundary-preserving projections, or Scope Geometry. Nevertheless, the motivation for introducing activation trajectories should now be apparent. The Representation Invariance Theorem (Theorem 9.10) established that semantic equivalence depends on admissible structural transformations rather than raw coordinate values, and activation trajectories provide the continuous objects from which such transformations may ultimately be extracted. Chapter 13 develops the mathematical machinery required for this step by constructing analytic signal representations of these trajectories, allowing phase relationships and synchronization phenomena to be studied directly.

12.7 Chapter Summary

This chapter reformulated transformer computation as a discrete dynamical system evolving through activation space. We introduced activation trajectories, layerwise Jacobians, local linearization, trajectory curvature, cumulative path length, local Lyapunov exponents, and principal tangent directions, each derived directly from standard linear algebra and dynamical systems theory. No spectral or wave-based interpretation has yet been introduced; the purpose here has been to establish a rigorous dynamical foundation on which the subsequent analytic signal framework may be constructed.

Status of Results: Chapter 12

Established background. Discrete dynamical systems, Jacobian matrices, singular value decomposition, Lyapunov analysis, differential geometry, and local linearization.

Formal developments. Activation trajectories, discrete trajectory curvature, trajectory length, local layerwise Lyapunov exponents, and their application to transformer residual streams.

Empirical hypotheses. Semantic organization within transformer models is reflected more faithfully by the geometry of activation trajectories than by isolated activation vectors. This hypothesis motivates the spectral analysis developed in Chapter 13.

Chapter 13

Resonant Analysis of Activation Trajectories

Every signal contains both amplitude and phase. Understanding often lies not merely in what changes, but in how changes remain synchronized.
— author's formulation

The previous chapter reformulated transformer computation as a discrete dynamical system whose principal mathematical object is the activation trajectory. The present chapter extends that analysis by constructing an analytic signal representation of activation trajectories using the discrete Hilbert transform, a construction well established throughout signal processing and communications theory, where analytic signals separate amplitude and phase components of a real signal. The novelty proposed here is not the mathematics of analytic signals themselves, but their application to transformer activation trajectories. Throughout this chapter, every reference to oscillation, resonance, standing waves, synchronization, or phase should be understood as an analytic description of latent trajectory geometry after transformation into analytic signal space; no claim is made that physical waves exist inside neural networks.

13.1 Motivation

Many existing interpretability techniques analyze activation magnitude, feature direction, neuron attribution, or attention weights, generally treating successive layer activations as independent observations or static geometric objects. Activation trajectories suggest a different viewpoint: once activations are interpreted as ordered signals evolving through layer depth, additional mathematical quantities become available, including instantaneous phase, phase synchronization, local frequency, amplitude envelopes, coherence, and spectral stability. These quantities are standard throughout signal analysis but have been only minimally explored within transformer representations.

13.2 Analytic Signal Representation

Consider an activation trajectory $\gamma = (\mathrm{x}_0, \mathrm{x}_1, \ldots, \mathrm{x}_L)$. Each coordinate defines a discrete real-valued signal indexed by layer depth; applying the discrete Hilbert transform produces a corresponding analytic representation.

Definition 13.1 (Analytic activation signal)**.** Let $\mathrm{x}_l \in \mathbb{R}^{d_{\text{model}}}$ denote the residual activation at layer l. The associated analytic signal is $\mathrm{z}_l = \mathrm{x}_l + i\,\mathcal{H}(\mathrm{x}_l)$, where $\mathcal{H}$ denotes the discrete Hilbert transform applied coordinate-wise.

The analytic signal contains the original activation as its real part and its Hilbert transform as the imaginary part, together defining a complex-valued trajectory $\mathrm{z}_0, \ldots, \mathrm{z}_L$ that will serve as the principal object of analysis throughout the remainder of this chapter.

13.3 Amplitude, Instantaneous Phase, and Synchronization

The analytic representation permits every activation to be decomposed into polar coordinates: the instantaneous amplitude $A_l = |\mathrm{z}_l|$ and the instantaneous phase $\phi_l = \arg(\mathrm{z}_l)$. Amplitude measures the local magnitude of the activation trajectory, while phase describes its relative orientation within analytic signal space; neither quantity individually characterizes semantic organization, but meaningful structure is hypothesized to emerge through the interaction of phase relationships across layers and tokens.

Signal processing frequently studies synchronization between multiple analytic signals, a construction we adapt to activation trajectories. Given two trajectories γ_i and γ_j, their instantaneous phase difference is $\Delta\phi_l = \phi_l^{(i)} - \phi_l^{(j)}$; perfect synchronization corresponds to constant phase difference, while rapidly varying phase indicates desynchronization. The principal synchronization statistic employed throughout this monograph is the Phase Locking Value (PLV), widely used in neuroscience and signal processing.

Definition 13.2 (Phase Locking Value)**.** Given N paired observations, $\text{PLV} = \left|\frac{1}{N}\sum_{k=1}^{N} e^{\,i\Delta\phi_k}\right|$.

The Phase Locking Value satisfies $0 \leq \text{PLV} \leq 1$: values approaching 1 indicate strong phase coherence, values near 0 phase dispersion, and within the present framework PLV is interpreted as a quantitative measure of synchronization between evolving activation trajectories.

The derivative of instantaneous phase defines the local frequency; since transformer layers are discrete we employ finite differences, $\omega_l = \phi_{l+1} - \phi_l$. Large values correspond to rapid phase evolution, small values to relatively stable semantic progression, and ω_l will later contribute to the detection of abrupt context transitions.

13.4 Semantic Coherence Hypothesis

Having introduced the principal spectral quantities, we state the central hypothesis motivating the remainder of Volume III.

Hypothesis 13.3 (Semantic coherence)**.** Activation trajectories associated with successful reasoning tasks exhibit higher phase coherence than trajectories associated with hallucination, context collapse, or inconsistent reasoning.

This hypothesis is intentionally empirical: unlike the theorems of Chapters 8–10, it is not proved mathematically, but motivates the experimental protocols of Chapter 15.

13.5 The Marine Operator

The previous quantities characterize local spectral properties. We now introduce a derived operator intended to summarize the overall spectral organization of an activation trajectory. Unlike the amplitude, phase, and PLV quantities above, which are standard signal-processing constructions applied to a new object, the Marine Operator is an original proposal of this monograph: a specific way of combining those quantities into a single filtering functional, motivated by acoustic and radar salience filtering.

Definition 13.4 (Marine Operator, abstract form)**.** Let γ be an activation trajectory. The Marine Operator $\mathcal{M}(\gamma)$ is any functional combining instantaneous amplitude, phase evolution, local synchronization, and trajectory stability into a single diagnostic operator $\mathcal{M} : \Gamma \to \mathbb{R}^k$, where Γ denotes the space of activation trajectories.

We now give one concrete instance of this abstract form, built from the quantities already introduced. Nothing in the remainder of this chapter depends on this being the only possible instance; it is presented as a candidate realization, to be evaluated experimentally in Chapter 15 alongside whatever alternatives later work proposes.

13.5.1 Doppler Velocity

The instantaneous frequency $\omega_l = \phi_{l+1} - \phi_l$ introduced above measures local phase change but says nothing about how that rate of change is itself evolving across depth. Borrowing the Doppler analogy from acoustic and radar signal processing, we track this second-order quantity directly.

Definition 13.5 (Doppler velocity)**.** Let ϕ_l be the unwrapped instantaneous phase at layer l, with $\text{wrap}(\phi) = \text{mod}(\phi + \pi, 2\pi) - \pi$ applied to keep successive differences in $(-\pi, \pi]$. The *Doppler velocity* is the discrete layerwise derivative of the unwrapped phase trajectory, $v_l = \text{wrap}(\phi_{l+1} - \phi_l) - \text{wrap}(\phi_l - \phi_{l-1})$.

Large $|v_l|$ indicates that the local frequency ω_l is itself changing rapidly — a trajectory undergoing acceleration through phase space rather than settling into a stable rate — which we take as a candidate indicator of representational restructuring beyond what ω_l alone captures.

13.5.2 The Salience Gate

We combine phase coherence and Doppler velocity into a single gate that attenuates layers exhibiting either low coherence or high Doppler drift.

Definition 13.6 (Marine salience gate)**.** Let PLV_l denote the Phase Locking Value computed in a window around layer l, and let v_l be the Doppler velocity there. The *Marine salience gate* is

$$G_l = \sigma_{\text{gate}}(\alpha \ \text{PLV}_l - \beta |v_l| - \gamma),$$

where $\sigma_{\text{gate}}(x) = 1/(1 + e^{-x})$ is the logistic sigmoid, $\alpha, \beta > 0$ weight the two contributions, and γ is a static attenuation threshold.

The gate satisfies $G_l \in (0, 1)$: layers with high coherence and low Doppler drift receive G_l near 1 and are passed through largely unchanged, while layers with low coherence or high drift receive G_l near 0 and are attenuated. Applying the gate to the original activation by $\tilde{\text{x}}_l = G_l \cdot \text{x}_l$, and then rescaling to restore the original norm, yields a filtered trajectory in which incoherent, rapidly drifting layers contribute less to downstream diagnostic statistics than stable, phase-locked ones.

13.5.3 Energy Conservation

Filtering by G_l changes the magnitude of x_l by construction. The following observation records that this magnitude change can always be undone by rescaling, so that the gate reshapes direction and relative emphasis across layers without discarding activation energy outright.

Proposition 13.7 (Energy conservation under rescaling)**.** *Let $E_l = \|\mathrm{x}_l\|_2^2$ and let $\hat{\mathrm{x}}_l = \kappa_l\, G_l\, \mathrm{x}_l$ with $\kappa_l = \|\mathrm{x}_l\|_2 / (\|G_l\, \mathrm{x}_l\|_2 + \epsilon)$ for small $\epsilon > 0$. Then $\|\hat{\mathrm{x}}_l\|_2 \to \|\mathrm{x}_l\|_2$ as $\epsilon \to 0$, so the rescaled filtered activation recovers the original energy E_l exactly in the limit.*

Proof. Since G_l is a positive scalar, $\|G_l \mathrm{x}_l\|_2 = G_l \|\mathrm{x}_l\|_2$. Substituting κ_l gives $\|\hat{\mathrm{x}}_l\|_2 = \kappa_l G_l \|\mathrm{x}_l\|_2 = \dfrac{\|\mathrm{x}_l\|_2}{\|\mathrm{x}_l\|_2 + \epsilon / G_l} \|\mathrm{x}_l\|_2$, which tends to $\|\mathrm{x}_l\|_2$ as $\epsilon \to 0$. □

This is a normalization convenience, not a deep result: it guarantees that applying the Marine filter does not, by itself, cause activation magnitudes to decay or explode across layers, which matters for using G_l as a diagnostic weight rather than a replacement for the original representation. We state it as a proposition rather than a theorem because its content is a routine renormalization identity, not a structural claim about admissibility.

The precise choice of α, β, γ, and the decision to treat $\mathcal{M}(\gamma)$ as $(G_0, \ldots, G_L)$ or as some summary statistic of it, are left as implementation choices for Chapter 14, which realizes this operator as part of the diagnostic pipeline.

13.6 Interpretive Remarks

Before proceeding, three observations bear on how these quantities should be read. None of the spectral quantities introduced here require the existence of physical oscillations inside transformer networks, since the Hilbert transform constructs an analytic representation of arbitrary signals regardless of their physical origin. Resonance should likewise be interpreted mathematically rather than metaphysically: within this monograph it denotes persistent synchronization patterns observed in transformed activation trajectories, a property of the analytic representation rather than a physical wave propagating through hardware. Finally, the quantities introduced here remain independent of Scope Geometry; only in Chapter 14 will admissibility-preserving projections connect these spectral measurements to the structural framework developed earlier in the book.

Where the Representation Invariance Theorem established that semantic equivalence depends on admissible transformations rather than internal coordinates, the present chapter has instead introduced mathematical quantities capable of describing how those internal coordinates evolve in the first place; these spectral quantities are therefore not themselves semantic invariants, but provide measurable observables from which admissibility may be inferred. Structural Semantics remains the theoretical framework, while analytic signal analysis supplies one possible family of empirical measurement operators.

13.7 Chapter Summary

This chapter introduced the analytic signal representation of activation trajectories. Using the discrete Hilbert transform, residual stream vectors were extended into complex-

valued trajectories possessing instantaneous amplitude, phase, local frequency, and synchronization properties, and these were combined into a concrete instance of the Marine Operator via the Doppler velocity and the salience gate G_l. The principal empirical proposal is that semantic coherence may be reflected by stable phase relationships between evolving activation trajectories; this proposal remains explicitly hypothetical and is subjected to experimental evaluation in Chapter 15. Before those experiments can be formulated, the spectral quantities developed here must be transformed into practical diagnostic algorithms, the objective of the following chapter.

Status of Results: Chapter 13

Established background. Analytic signals, Hilbert transforms, instantaneous phase, amplitude envelopes, Phase Locking Values, and classical signal-processing methods.

Formal developments. The analytic activation signal, discrete instantaneous frequency, application of Phase Locking Values to activation trajectories, the abstract definition of the Marine Operator together with a concrete instance built from the Doppler velocity and salience gate, and the energy-conservation renormalization proposition.

Empirical hypotheses. Semantic coherence is associated with increased phase synchronization between activation trajectories, and the salience gate's attenuation pattern is associated with regions of representational instability. These hypotheses are tested in the experimental framework developed in Chapters 14 and 15.

Chapter 14

Transformer Diagnostics and Algorithmic Architecture

A mathematical theory becomes scientifically useful only when its quantities can be computed.
— adapted from Lord Kelvin

The preceding chapter introduced the spectral quantities forming the empirical core of the proposed analytic framework, but did not specify how these quantities should be extracted from real transformer models. The objective of the present chapter is therefore computational rather than theoretical: we construct an algorithmic architecture for estimating the quantities of Chapter 13 directly from transformer activation streams. Throughout this chapter we remain deliberately implementation-neutral; the algorithms described here are intended to be realizable in PyTorch, JAX, TensorFlow, or equivalent numerical environments; reference implementations are left for future companion material rather than included in this monograph.

14.1 Diagnostic Pipeline

The proposed diagnostic framework consists of six sequential stages: extraction of residual stream activations; construction of activation trajectories; analytic signal transformation; spectral feature computation; admissibility estimation; and diagnostic reporting. Each stage operates independently of any particular language model, consequently the framework may be applied to open-weight transformer architectures regardless of parameter count or tokenizer design.

For a transformer model M consisting of L layers and an input sequence $X = (x_1, \dots, x_T)$, forward evaluation produces the residual tensor $\mathrm{R} \in \mathbb{R}^{(L+1)\times T\times d_{\text{model}}}$, whose elements $\mathrm{r}_{l,t}$ record the residual activation associated with token t after completion of layer l; this tensor forms the raw observational data for all subsequent diagnostic computations. For each token t we define the activation trajectory $\gamma_t = (\mathrm{r}_{0,t}, \mathrm{r}_{1,t}, \dots, \mathrm{r}_{L,t})$, and the collection $\Gamma = \{\gamma_1, \dots, \gamma_T\}$ contains one trajectory for every token within the context window; subsequent computations operate exclusively on these trajectories rather than directly on the residual tensor.

Each activation trajectory undergoes analytic transformation using the construction of Chapter 13. For every layer l, the amplitude envelope A_l, instantaneous phase ϕ_l,

instantaneous frequency ω_l, local phase difference $\Delta\phi_l$, Phase Locking Value, and trajectory curvature κ_l are computed, collectively defining a spectral descriptor vector s_t for every token trajectory.

Definition 14.1 (Diagnostic state vector)**.** For every activation trajectory γ_t, define $\mathrm{d}_t = (A, \phi, \omega, \mathrm{PLV}, \kappa, \lambda)$, where λ denotes the local Lyapunov statistic of Chapter 12.

The diagnostic state vector summarizes the local dynamical behavior of an activation trajectory; subsequent statistical procedures operate on d_t rather than on individual observables.

14.2 Admissibility Estimation and Context Collapse

The central objective of the diagnostic framework is not merely to compute spectral quantities but to estimate semantic admissibility.

Definition 14.2 (Admissibility estimator)**.** An admissibility estimator is a mapping $\mathcal{A} : \mathrm{d} \to [0, 1]$ assigning each diagnostic state vector an estimated probability of admissible semantic evolution.

The precise form of $\mathcal{A}$ is intentionally left unspecified. Different implementations may employ probabilistic classifiers, Bayesian estimators, neural predictors, or deterministic threshold rules, allowing the mathematical framework developed in earlier chapters to remain independent of particular machine-learning techniques. The salience gate G_l of Chapter 13 is one such candidate: setting $\mathcal{A}(\mathrm{d}_t)$ to the layerwise average of G_l over a token's trajectory yields a fully specified, if simple, admissibility estimator built entirely from quantities already defined; nothing in the remainder of this chapter depends on that particular choice, and the experiments of Chapter 15 evaluate it against the alternatives listed there. Whichever estimator is chosen, treating $\mathcal{A}(\mathrm{d}_t)$ as a probability of admissible evolution presupposes a formal measure of admissibility over semantic space in the first place; Appendix G constructs exactly such a measure, together with the entropy and mutual-information quantities that a more statistically grounded estimator than the salience gate might use in its place.

Definition 14.3 (Context collapse)**.** A context collapse event occurs whenever $\mathcal{A}(\mathrm{d}_t) < \tau$ for some admissibility threshold $\tau \in (0, 1)$.

The threshold τ is a hyperparameter whose optimal value depends on the application domain. Rather than identifying isolated incorrect tokens, this definition identifies regions of semantic instability within the evolving activation trajectory.

14.3 Computational Complexity

The proposed diagnostic framework introduces relatively modest additional computational cost. Residual stream extraction requires no additional forward passes, trajectory construction is linear in both sequence length and transformer depth, and analytic signal computation using the Fast Fourier Transform requires $O(L \log L)$ operations per activation coordinate. The overall complexity of the diagnostic pipeline is therefore $O(LTd_{\text{model}})$ up to logarithmic factors associated with Hilbert transform implementation, scaling linearly with the size of the activation tensor.

14.4 Case Study: Multi-Head Attention Residuals

The preceding section compared the diagnostic architecture of this chapter with existing interpretability techniques at the level of purpose. It is also useful to examine a single recent architectural proposal in enough detail to see the Scope Calculus vocabulary of Chapters 6–9 applied to something concrete. This section reads Luo et al. [34] through that vocabulary. The reading is offered as an interpretation, not as a formal result: nothing here constructs an admissibility-preserving projection in the sense of Definition 9.7 and verifies its hypotheses, so no claim is made that the architecture instantiates Scope in the way Theorem 9.10 would require. The value of the exercise is illustrative — it shows what the Scope Calculus vocabulary buys a reader confronted with an ordinary architecture paper, and where that vocabulary runs out.

14.4.1 The Architecture

An ordinary transformer residual stream accumulates layer outputs by addition, $h_\ell = h_{\ell-1} + f_\ell(h_{\ell-1})$, so that an earlier contribution s_i remains causally present in h_ℓ but is no longer separately addressable: it has been folded into a running sum rather than retained as a distinguishable object. Attention Residuals [34] relax this by keeping the sequence of prior layer outputs $S_\ell = (s_0, \dots, s_\ell)$ available and retrieving from it via a learned softmax query, so that each layer reads a weighted combination $\sum_i \alpha_i s_i$ of its full depth history rather than only its immediate predecessor. Multi-Head Attention Residuals (MHAR) then observe that this retrieval query is shared across the entire representational width: every feature subspace of h_ℓ is forced to read the same weighted combination α of the history, even when different subspaces would prefer to weight the same history differently. MHAR reshapes the single query into H per-subspace queries, giving each subspace its own softmax over the depth history, at essentially no additional parameter or compute cost; the ordinary Attention Residuals architecture is recovered exactly as the special case $H = 1$. Trained from scratch across several model scales, MHAR improves validation loss over both the plain residual stream and single-query Attention Residuals, with the improvement reported to grow with model width.

14.4.2 Reading the Architecture as Scope Retrieval

The depth history $S_\ell = (s_0, \dots, s_\ell)$ available at retrieval time reads naturally as an admissible source family in the sense of Chapter 6: at each layer, the retrieval operation must select from among several available prior states rather than being handed a single determined predecessor, much as a child scope's Interior consists of several admissible states rather than one. The retrieval itself, a weighted combination followed by use in the next sublayer, matches the two-step pattern of $\mathcal{B}_d$ followed by $\mathcal{C}_l$ already used in Appendix C to interpret function application: the history is first bound into a candidate combination, and that combination is then collapsed into the single vector the next layer consumes.

Read this way, ordinary Attention Residuals apply one $\mathcal{C}_l$ operation across the entire width of h_ℓ: a single admissible combination is selected, and every subspace of the output is constructed from that same combination. MHAR instead applies H separate collapse operations, one per subspace, each free to select a different admissible combination of the same history. Chapter 7 already distinguishes a single global admissibility check from a factorized one; MHAR is a concrete instance of the latter, applied not to the primitive alphabet itself but to the routing mechanism built on top of it.

14.4.3 What the Reading Clarifies, and What It Does Not

The reading clarifies why the single-query architecture pays an increasing cost as width grows: forcing every subspace through one $\mathcal{C}_l$ is, in the vocabulary of this monograph, an admissibility restriction that becomes more binding as the number of genuinely different subspaces increases, since more of them are made to accept a combination that was not built for them specifically. It also clarifies why $H = 1$ recovering the original architecture exactly is the right design choice from this monograph's perspective: it is the discipline, familiar from the zero-initialized gates discussed in Chapter 1's distinction between admissible and inadmissible transitions, of introducing a structural change as an identity transformation before allowing it to diverge.

The reading does not establish that MHAR's retrieval operators satisfy the boundary-preservation conditions of Definition 9.7, nor that the Representation Invariance Theorem applies to a comparison between the single-query and multi-query architectures. Doing so would require specifying what boundary constraints the depth history is required to preserve and verifying that both retrieval mechanisms respect them, which is an empirical and architectural question this monograph does not address. The case study should therefore be read as showing that an independently developed, empirically motivated architecture exhibits the same qualitative shape — a single shared collapse operation restricting what a heterogeneous system can express, relaxed by factorizing that collapse per subspace — that Chapters 6–9 develop abstractly, not as evidence for any theorem in this monograph.

14.5 Relationship to Existing Interpretability Methods

The proposed diagnostic architecture complements rather than replaces existing interpretability techniques: attention visualization identifies communication pathways between tokens, activation patching identifies causal computational circuits, sparse autoencoders identify approximately monosemantic features, and logit attribution estimates token-level prediction contributions, whereas the present framework analyzes the global dynamical evolution of representations. These approaches are therefore largely orthogonal and may be combined within a unified interpretability workflow.

14.6 Limitations

Several limitations should be emphasized. The diagnostic state vector provides descriptive rather than causal information — high phase synchronization does not by itself imply successful reasoning. Admissibility estimation depends on the quality of the chosen estimator $\mathcal{A}$, and poor statistical estimation may produce unreliable diagnostic results even if the underlying mathematical framework is correct. Different transformer architectures may also exhibit substantially different spectral statistics despite similar downstream behavior, so normalization procedures will likely be required before cross-model comparisons become meaningful. These limitations motivate the controlled experimental framework developed in the following chapter.

14.7 Chapter Summary

This chapter translated the mathematical quantities of Chapter 13 into a practical computational pipeline. Beginning with residual stream extraction, activation trajectories were transformed into analytic signals, summarized through diagnostic state vectors, and mapped to estimated admissibility scores capable of identifying potential regions of semantic instability. None of these procedures establishes the empirical validity of the proposed framework; they specify only how the relevant measurements may be computed. The scientific evaluation of these measurements requires carefully controlled experiments, statistical hypothesis testing, and explicit falsification criteria, the subject of the next chapter.

Status of Results: Chapter 14

Established background. Residual stream extraction, computational complexity analysis, signal processing algorithms, statistical feature extraction, and the Attention Residuals and Multi-Head Attention Residuals architectures of Luo et al. [34].

Formal developments. The diagnostic pipeline, diagnostic state vector, admissibility estimator, and formal definition of context collapse as an admissibility-threshold event.

Interpretive discussion. §14.4 reads Multi-Head Attention Residuals through the Scope Calculus vocabulary of Chapters 6–9, as an illustration rather than a theorem: no admissibility-preserving projection is constructed for it, and no claim of the monograph depends on the reading.

Empirical hypotheses. Diagnostic quantities derived from activation trajectories can identify regions of semantic instability prior to observable output failure. The validity of this hypothesis is evaluated experimentally in Chapter 15.

Chapter 15

Experimental Predictions, Controls, and Falsification Protocols

The value of a scientific theory lies not merely in what it explains, but in what observations could demonstrate it to be incorrect.
— Karl Popper (adapted)

The preceding chapters introduced the mathematical and computational framework of Structural Semantics for transformer analysis. The present chapter addresses the central scientific question: can these proposed quantities be shown to possess explanatory power beyond existing interpretability methods? Unlike previous chapters, which primarily developed mathematical structures, this chapter develops an empirical research program in which every hypothesis is explicitly falsifiable. Failure of the stated predictions should be interpreted as evidence against the proposed framework rather than evidence requiring ad hoc modification.

15.1 Scientific Methodology

The empirical program developed in this monograph follows four guiding principles: every proposed diagnostic quantity must be measurable using existing transformer architectures; every hypothesis must admit explicit statistical rejection; comparisons must be performed against established baseline methods; and negative results are regarded as scientifically valuable because they delimit the scope of applicability of the proposed framework. These principles reflect the methodological distinction introduced in Chapter 1 between formal mathematics and empirical science.

15.2 General Experimental Design

Let $\mathcal{M} = \{M_1, \dots, M_K\}$ denote a collection of transformer models. Experiments are conducted over a benchmark collection $\mathcal{D} = \{D_1, \dots, D_N\}$ containing reasoning tasks, mathematical proofs, code generation, multi-document question answering, and long-context inference problems. For every prompt $x \in \mathcal{D}$, each model produces output tokens, residual stream activations, activation trajectories, diagnostic state vectors, and admissibility estimates, forming the dataset used throughout the remainder of the chapter.

15.3 Hypothesis I: Phase Coherence and Semantic Stability

Hypothesis 15.1 (Phase coherence hypothesis)**.** Successful reasoning trajectories exhibit larger average Phase Locking Values than trajectories containing semantic inconsistency or hallucination: $\mathrm{PLV}_{\text{correct}} > \mathrm{PLV}_{\text{incorrect}}$.

A benchmark of at least $N \geq 1000$ paired reasoning traces should be constructed, each independently classified by human annotators as correct, partially correct, hallucinated, or internally inconsistent, and average Phase Locking Values then computed for each category. The null hypothesis is $H_0 : \mu_{\text{correct}} = \mu_{\text{incorrect}}$ against the alternative $H_1 : \mu_{\text{correct}} > \mu_{\text{incorrect}}$; depending on the empirical distribution, Student's t-test, Welch's test, or the Mann–Whitney U test may be employed, with statistical significance assessed throughout this monograph at $\alpha = 0.01$. The hypothesis is rejected whenever $p > 0.01$, or when the observed effect size is practically negligible despite statistical significance.

15.4 Hypothesis II: Frequency Drift and Context Collapse

Hypothesis 15.2. Abrupt semantic inconsistency is accompanied by statistically detectable changes in instantaneous frequency.

Let ω_l denote the instantaneous frequency of Chapter 13; large discontinuities $|\omega_{l+1} - \omega_l|$ are hypothesized to precede observable failures of reasoning. Long-context reasoning tasks are modified through controlled insertion of contradictory premises, and the resulting activation trajectories are analyzed before, during, and after the injected contradiction and compared with ordinary successful reasoning traces. If no statistically significant frequency differences are observed between successful and contradictory reasoning trajectories, the hypothesis is rejected.

15.5 Hypothesis III: Admissibility and Error Prediction

Hypothesis 15.3. Declining admissibility estimates precede observable output errors.

Rather than identifying hallucinations after incorrect tokens have already appeared, admissibility estimation is hypothesized to detect semantic instability during internal computation. For every generated sequence, the first observable output error is recorded, diagnostic quantities computed before that token are analyzed, and receiver operating characteristic curves and precision–recall statistics are calculated to determine predictive performance, summarized using the area under the ROC curve, precision, recall, the F_1 score, and the Matthews Correlation Coefficient.

Example 15.4 (What this hypothesis does and does not claim)**.** Suppose the experiment above returns a strong result: admissibility estimates decline detectably several layers before an incorrect token is sampled, with an area under the ROC curve well above chance. This would establish, for the specific models and tasks tested, that the diagnostic quantities of Chapter 14 carry predictive information about upcoming output errors — an empirical finding, reported with the statistical qualifications Chapter 15 requires throughout. It would not, by itself, establish a working intervention: nothing in this chapter specifies what a system should do upon detecting declining admissibility, whether halting, revising, or continuing is preferable in a given context, or what computational overhead an intervention would add beyond the detection step itself. A system

that halts and backtracks on a declining-admissibility signal is a natural extension to propose, but it is a distinct research question — one requiring its own experimental validation and its own comparison against existing decoding strategies — not a consequence already established by Hypothesis III.

15.6 Control Experiments

Perhaps the greatest danger facing any new diagnostic quantity is that it merely reproduces information already contained in simpler statistics, so every experiment above must be accompanied by control analyses. Phase coherence must be distinguished from token confidence: softmax entropy is computed alongside PLV, and regression analysis determines whether Phase Locking Values retain predictive power after controlling for entropy. Large activation magnitudes might artificially influence spectral statistics, so activation vectors are normalized before analytic transformation and equivalent experiments are repeated using raw, normalized, and randomized activations. To determine whether synchronization reflects genuine structure rather than statistical coincidence, phase values are randomly permuted while preserving amplitude and the diagnostic framework re-evaluated, with significant performance degradation supporting the interpretation that phase relationships contain meaningful information. Finally, experiments should be repeated across transformer families differing in parameter count, tokenizer, training corpus, attention implementation, and instruction tuning, since agreement across architectures would strengthen the claim that the observed phenomena reflect general computational organization rather than architecture-specific artefacts.

15.7 Baseline Comparisons

The proposed framework should not be evaluated in isolation, but compared against established diagnostic approaches including attention entropy, logit confidence, activation norm, perplexity, sparse autoencoder feature activation, activation patching methods, and uncertainty estimation. A proposed diagnostic quantity possesses scientific value only if it provides predictive information beyond these existing baselines.

15.8 Failure Modes

Several outcomes would argue against the proposed framework: Phase Locking Values showing no predictive relationship with semantic correctness; instantaneous frequency exhibiting no systematic relationship with context collapse; admissibility estimates failing to predict errors before observable output failure; existing baseline methods consistently outperforming every proposed spectral quantity; or performance failing to generalize across transformer architectures. Observation of any of these outcomes should be interpreted as evidence requiring revision or rejection of the corresponding hypotheses.

15.9 Discussion

The purpose of the present chapter is not to demonstrate that the proposed framework succeeds, but to specify the conditions under which success or failure may be determined objectively. This distinction is essential: the mathematical framework developed

in Volumes I and II remains valid independently of these experiments, whereas the empirical usefulness of the diagnostic operators of Volume III depends entirely on experimental evidence. The separation between formal mathematics and empirical evaluation therefore remains explicit throughout the monograph.

15.10 Chapter Summary

This chapter established a complete experimental program for evaluating the diagnostic framework proposed throughout Volume III. Three principal hypotheses concerning phase coherence, frequency drift, and admissibility prediction were formulated together with statistical procedures, control experiments, baseline comparisons, and explicit falsification criteria. The completion of this empirical program concludes the technical development of Structural Semantics; the remaining volume returns to a broader perspective, comparing the framework with neighboring fields, identifying its limits, and discussing future directions.

Status of Results: Chapter 15

Established background. Experimental design, statistical hypothesis testing, effect sizes, receiver operating characteristic analysis, and benchmark evaluation.

Formal developments. Explicit experimental protocols, statistical controls, baseline comparisons, and falsification criteria for evaluating the proposed diagnostic framework.

Empirical hypotheses. Phase synchronization, instantaneous frequency, and admissibility estimation provide measurable indicators of semantic coherence whose predictive value exceeds existing baseline diagnostics. These hypotheses remain subject to experimental confirmation or rejection.

Part IV

Synthesis

Chapter 16

Cross-Disciplinary Integration and Comparative Landscape

Science progresses not merely by constructing new theories, but by understanding precisely how those theories relate to existing ones.
— author's formulation

The preceding three volumes developed Structural Semantics from first principles. The purpose of the present chapter is fundamentally different: rather than introducing additional mathematical structures, we situate Structural Semantics within the broader scientific landscape. Throughout this chapter our objective is comparison rather than competition; Structural Semantics is not presented as replacing existing frameworks, but as addressing a different class of questions through a different mathematical language. To facilitate comparison, we evaluate each framework according to four questions: what constitutes a semantic object, how is semantic change represented, how is correctness evaluated, and what mathematical structures are fundamental.

16.1 Predictive Processing and Active Inference

Predictive Processing and Active Inference describe cognition as the continuous minimization of prediction error under hierarchical generative models [11, 18]. Structural Semantics shares with this tradition an emphasis on dynamics over static representation and on hierarchical organization, but its mathematical objectives differ: Predictive Processing centers on probabilistic inference and free-energy optimization, while Structural Semantics studies admissibility-preserving transformations independent of probabilistic interpretation, with prediction error playing no primitive role in the Scope Calculus. Consequently the two approaches should be regarded as complementary, one emphasizing optimization and the other structural constraint.

16.2 Mechanistic Interpretability

Mechanistic interpretability attempts to explain neural computation by identifying internal circuits responsible for observable model behavior [15, 42, 44]. Structural Semantics aligns closely with this program in seeking internal mathematical explanations rather

than behavioral descriptions, but the two frameworks operate at different levels of description: circuit analysis identifies computational components such as neurons, attention heads, and sparse features, while Scope Geometry analyzes the global organization of admissible trajectories connecting those components.

16.3 Information Geometry

Information geometry studies statistical models using differential geometric methods, introducing Fisher information, geodesics, divergence functions, and statistical manifolds [1]. Both frameworks employ manifold language and investigate geometric organization, but information geometry derives its structure from probability distributions while Scope Geometry derives its structure from admissibility constraints, so the corresponding metrics encode fundamentally different objects — statistical distinguishability in one case, preservation of contextual organization in the other.

16.4 Category Theory

Category theory provides one of the principal mathematical languages of this monograph, though Structural Semantics employs only a deliberately restricted subset of categorical machinery: objects represent bounded scope geometries and morphisms represent admissible context-preserving transformations, with no assumptions regarding limits, exponentials, adjunctions, enriched categories, or higher categories unless explicitly introduced. This structural minimalism distinguishes the present framework from category-theoretic approaches seeking maximal abstraction; category theory functions here primarily as an organizational language for admissible transformations.

16.5 Formal Language Theory

Formal language theory studies symbolic computation through grammars, automata, rewrite systems, and logical calculi. Structural Semantics inherits several ideas from this tradition: the primitive alphabet $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$ functions analogously to a rewriting alphabet, canonical normal forms resemble rewriting theory, and admissible transformations resemble typed derivations. The principal distinction concerns semantics: formal language theory primarily studies syntactic derivability, while Structural Semantics studies preservation of contextual admissibility, so that syntax becomes one particular instance of a broader structural framework.

16.6 Classical Structural Semantics

The terminology adopted in this monograph inevitably overlaps with the classical linguistic tradition known as Structural Semantics [12, 36], discussed at length in Chapter 2. Both approaches emphasize relationships rather than isolated entities and reject purely referential theories of meaning, but the mathematical scope differs substantially: classical structural semantics studies relationships between linguistic units, while the present work studies admissible transformations within arbitrary computational systems, so the relationship should be understood as one of historical continuity rather than mathematical identity.

16.7 Process Philosophy

Chapter 10 established a formal translation functor between Whiteheadian process ontology and Scope Geometry, demonstrating that many operational aspects of process philosophy admit structural reformulation. Structural Semantics neither adopts nor rejects Whitehead's metaphysical commitments, identifying instead those components capable of operational formalization; Subjective Form remained outside the image of the translation functor, marking the precise boundary between phenomenological philosophy and structural mathematics.

16.8 Large Language Models

Modern large language models constitute the primary computational domain investigated throughout Volume III, but Structural Semantics should not be regarded as a theory of transformers alone. The Representation Invariance Theorem established that semantic organization depends on admissible transformations rather than specific architectures, so nothing in principle restricts the framework to transformers; recurrent neural networks, graph neural networks, state-space models, symbolic systems, and hybrid neuro-symbolic architectures may equally be analyzed whenever appropriate admissibility-preserving projections can be constructed.

16.9 Synthesis

Comparison across these diverse frameworks reveals a common pattern: every approach identifies an important aspect of semantic organization — Predictive Processing emphasizes optimization, mechanistic interpretability internal computation, information geometry statistical structure, formal language theory symbolic derivation, classical structural semantics lexical relations, and process philosophy becoming. Structural Semantics contributes a different perspective, whose central object is neither probability, syntax, prediction, consciousness, nor optimization, but the preservation of admissible structural transformations under changing representations, occupying a distinct position within the broader landscape of semantic theory.

16.10 Chapter Summary

This chapter situated Structural Semantics within the wider scientific and philosophical literature. Rather than competing with existing approaches, the framework was shown to address a complementary class of questions centered on admissible structural transformation. The next chapter explicitly enumerates the boundary conditions, scope limits, and non-claims of Structural Semantics, clarifying exactly what the framework does, and does not, attempt to explain.

Status of Results: Chapter 16

Established background. Predictive Processing, Active Inference, mechanistic interpretability, information geometry, formal language theory, classical structural semantics, category theory, and process philosophy.

Formal developments. Comparative positioning of Structural Semantics relative to existing frameworks and clarification of its distinctive mathematical focus.

Empirical hypotheses. None. This chapter is comparative and interpretive rather than mathematical or experimental.

Chapter 17

Scope Limits, Boundary Conditions, and Non-Claims

The strength of a mathematical theory is measured not only by what it explains, but equally by what it deliberately leaves unexplained.
— author's formulation

Every mathematical theory possesses boundaries, and one of the distinguishing characteristics of mature scientific frameworks is the explicit recognition of those questions lying beyond their intended domain of applicability. The purpose of this chapter is therefore not to weaken the preceding results but to strengthen their interpretation by specifying exactly what Structural Semantics does and does not claim. Throughout, the distinction introduced in Chapter 1 between formal mathematics, empirical hypotheses, and philosophical interpretation remains strictly enforced.

17.1 The Scope of Structural Semantics

Structural Semantics is fundamentally a theory of admissible transformations. Its primary mathematical object is neither consciousness, intelligence, meaning in the ordinary linguistic sense, nor biological cognition, but the preservation of structural organization under transformations constrained by explicit boundary relations. Accordingly, every theorem proved throughout Volumes I–III concerns admissibility, context preservation, structural stability, categorical composition, representation invariance, activation trajectory geometry, or diagnostic observables; no theorem proved in this monograph establishes any claim beyond these domains.

17.2 Consciousness, Truth, Agency, and Ethics

Structural Semantics makes no claim that systems possessing high semantic capacity are conscious, nor that conscious systems necessarily possess maximal structural semantic organization; phenomenal consciousness concerns subjective experience, while Structural Semantics concerns admissible transformations, and whether these domains ultimately intersect remains an open philosophical question outside the mathematical scope of the present framework.

Proposition 17.1 (Non-claim: consciousness)**.** *None of the theorems established in Chapters 6–15 implies the existence, absence, measurement, simulation, or explanation of phenomenal consciousness.*

Proof. Immediate. Neither phenomenal predicates nor subjective experience appear among the primitive symbols of the Scope Calculus, so no theorem concerning these concepts can be derived within the formal system. □

The Scope Calculus likewise evaluates structural coherence rather than factual accuracy: a perfectly coherent fictional narrative may satisfy every admissibility constraint while describing events that never occurred, while a scientifically correct explanation may violate contextual constraints if contradictory assumptions are introduced during reasoning, so truth and structural coherence remain logically independent, reflecting the separation introduced in the Disentanglement Operator (Definition 1.6). Nor does the framework provide a theory of agency: it neither attributes nor denies goals, intentions, preferences, values, autonomy, or moral status, and any computational system capable of executing admissible transformations may be analyzed within the Scope Calculus regardless of whether it is autonomous, externally controlled, deterministic, stochastic, biological, or mechanical. Nor does anything within the Scope Calculus determine whether a computational system ought to perform a particular action: mathematical admissibility is distinct from ethical acceptability, and an admissible transformation may be morally undesirable just as an ethically desirable action may require violating previously established contextual assumptions, so questions of ethics require additional normative frameworks not developed in this monograph.

17.3 General Intelligence and Biological Realism

The framework should not be interpreted as a definition of general intelligence: high structural semantic capacity does not imply competence across every domain, and systems may achieve impressive task performance while remaining structurally fragile, since general intelligence depends on many additional considerations including learning, abstraction, planning, memory, transfer, and adaptation, of which Structural Semantics addresses only one component. Nor is Structural Semantics intended as a biological theory: although examples throughout the text frequently reference neural networks and human cognition, no claim is made that the brain computes using the primitive alphabet $\{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$, or that cortical dynamics implement the Scope Calculus literally; the framework should instead be interpreted as an abstract mathematical language capable of describing admissible organization across multiple possible substrates.

17.4 Completeness and Boundary Conditions

No mathematical theory describing complex adaptive systems should be expected to explain every aspect of semantic behavior, and the present framework is intentionally incomplete: future extensions may incorporate probabilistic admissibility, temporal logics, multimodal reasoning, stochastic scope evolution, higher categorical structure, and continuous boundary fields, none of which are developed in the present monograph.

The formal results proved throughout this work depend on explicit assumptions, among the most important being the existence of bounded scope geometries, admissibility-preserving transformations, well-defined primitive operations, canonical normal forms,

admissibility-preserving projection maps where required, and finite computational histories for reconstruction results. Violation of these assumptions invalidates the corresponding theorems, so every application of Structural Semantics should begin by verifying that these hypotheses are satisfied.

17.5 Research Programme Versus Finished Theory

It is important to distinguish between those components already established mathematically and those proposed for future empirical investigation: Volumes I and II primarily developed formal mathematics, while Volume III introduced computational diagnostics together with explicit experimental hypotheses whose value only empirical investigation can determine. Accordingly, this monograph should be understood simultaneously as a completed mathematical framework and an incomplete empirical research program, and these two aspects should not be conflated.

17.6 Discussion

Scientific theories often become controversial not because of their formal content but because claims exceeding that content are attributed to them. Considerable effort has therefore been devoted throughout this monograph to separating mathematical theorem from philosophical interpretation and empirical hypothesis. This distinction is methodological rather than merely stylistic: readers may accept the formal mathematics while remaining skeptical of the empirical proposals, and future experiments may invalidate portions of Volume III without affecting the mathematical consistency of the Scope Calculus developed in Volumes I and II. Such independence is a strength rather than a weakness.

17.7 Chapter Summary

This chapter explicitly identified the limits of Structural Semantics. The framework does not claim to explain consciousness, truth, agency, ethics, or general intelligence, and provides neither a comprehensive theory of cognition nor a philosophy of mind; instead it contributes a formal mathematical language for describing admissible structural transformations together with an empirical program for investigating their computational realization. Having established these boundaries, the monograph concludes with a final synthesis of its principal mathematical, computational, and philosophical contributions.

Status of Results: Chapter 17

Established background. The importance of explicit scope conditions, methodological restraint, and boundary assumptions in mathematical and scientific theories.

Formal developments. Formal clarification of the domain of applicability of Structural Semantics together with explicit non-claims concerning consciousness, truth, agency, ethics, biological realism, and general intelligence.

Empirical hypotheses. None. This chapter establishes interpretive limits rather than new scientific predictions.

Chapter 18

Conclusion: The Architecture of Structural Semantics

Mathematics does not close inquiry. It gives inquiry a structure within which to continue.
— author's formulation

The central objective of this monograph has been to investigate whether semantic organization can be described independently of particular computational substrates, biological mechanisms, or philosophical commitments concerning consciousness. Beginning with this question, we developed a mathematical framework grounded in admissibility, structural preservation, and bounded scope geometries. The resulting theory, Structural Semantics, proposes that semantic organization is fundamentally a property of admissible transformations rather than a consequence of any particular physical implementation.

18.1 From Philosophy to Mathematics

The opening chapters began with an epistemic clarification. Discussions of artificial intelligence frequently employ the term "understanding" while referring to multiple distinct concepts including phenomenal consciousness, factual correctness, intentional agency, linguistic fluency, and contextual reasoning. Chapter 1 introduced the Disentanglement Operator to separate these concepts into independent domains, permitting the construction of a mathematical theory without requiring prior resolution of longstanding philosophical debates concerning the nature of consciousness or intentionality. The objective was therefore modest but precise: rather than explaining mind itself, the theory sought to characterize one particular aspect of intelligent systems, the preservation of structural context under transformation.

18.2 The Scope Calculus

The central mathematical contribution of the monograph is the Scope Calculus. Beginning with bounded scope geometries, primitive scope operations, boundary tensors, and admissibility relations, we developed a minimal algebra capable of describing context-preserving computation. Subsequent chapters demonstrated that these primitive operations admit categorical organization, canonical normal forms, and rigorous rewriting

properties, and Representation Invariance established that semantic organization depends on admissible structural transformations rather than particular representational substrates. Collectively these results provide the formal mathematical foundation of Structural Semantics.

18.3 Geometry of Semantic Organization

Traditional semantic theories frequently emphasize symbols, logical propositions, lexical relations, or statistical associations. Structural Semantics instead approaches semantic organization through geometry: context is represented by bounded scope geometries, semantic evolution becomes a sequence of admissible transformations, reasoning corresponds to trajectories constrained by boundary relations, and correctness becomes preservation of admissibility rather than merely production of plausible outputs. This geometric perspective permits methods from topology, category theory, algebra, and dynamical systems to contribute naturally to the study of semantic organization.

18.4 Computational Realization

Volume III demonstrated one possible computational realization of the abstract mathematical framework: transformer residual streams were interpreted as activation trajectories evolving through high-dimensional spaces, analytic signal methods extended these trajectories into complex-valued representations possessing amplitude, phase, synchronization, and frequency, and diagnostic algorithms were proposed for estimating admissibility from observable activation dynamics. Importantly, these computational constructions should not be confused with the mathematical foundations developed earlier: the Scope Calculus remains substrate independent, and the transformer framework represents one application rather than the definition of Structural Semantics itself.

18.5 Scientific Status

Throughout the monograph, considerable effort has been devoted to maintaining clear distinctions among formal mathematical theorems, proved deductively within the Scope Calculus; computational algorithms derived from those mathematical structures; and empirical hypotheses requiring experimental evaluation. The mathematical validity of the Scope Calculus does not depend on the success of the experimental program proposed in Volume III, while successful empirical results would support the usefulness of the computational framework without altering the formal proofs presented earlier. This separation between mathematics and empirical science is one of the guiding principles of the entire work.

18.6 Relationship to Existing Research

Structural Semantics occupies an intermediate position between several established disciplines, incorporating ideas from topology, category theory, formal language theory, information geometry, signal processing, and mechanistic interpretability while remaining reducible to none of them. The resulting framework should be understood as a synthesis rather than a replacement, whose objective is to provide a common mathematical

language capable of relating diverse approaches to semantic organization through the shared concept of admissible transformation.

The preceding chapters developed the conceptual architecture of Structural Semantics: the epistemic framing that separates structural capacity from phenomenology, the Scope Calculus that formalizes admissible transformation, and the diagnostic apparatus that applies it to transformer representations. The appendices supplied the supporting mathematical machinery this architecture was built on rather than material set aside from it — lambda calculus and abstract rewriting theory in Appendices A and B, categorical semantics in Appendices C and D, topology and sheaf theory in Appendices E and F, and measure-theoretic and probabilistic foundations in Appendix G, together with the collected proofs of Appendix I. Each was invoked at the point in the main text where its results became relevant rather than gathered as an afterthought, and together they are intended to show that the framework rests on a unified mathematical foundation rather than a collection of loosely related techniques assembled for the occasion.

18.7 Limitations and Future Directions

Structural Semantics does not provide a theory of consciousness, ethics, intentionality, biological cognition, or general intelligence, nor does it claim that transformer activation trajectories literally constitute physical waves or that analytic signal methods uniquely capture semantic organization; these questions remain outside the formal scope of the present work, and recognition of these limits strengthens rather than weakens the mathematical precision of the framework itself.

The present work opens numerous avenues for future research. From the mathematical perspective, extensions to stochastic scope geometries, continuous admissibility fields, enriched categories, and higher-dimensional rewriting systems appear particularly promising. From the computational perspective, large-scale empirical evaluation of activation trajectory diagnostics remains an immediate priority, with further opportunities in multimodal foundation models, autonomous agents, program synthesis, scientific reasoning, formal verification, and hybrid symbolic–neural architectures. More broadly, the framework suggests that admissibility may provide a common mathematical language for analysing structured computation across a wide variety of domains.

18.8 What Structural Semantics Does Not Claim

Chapter 17 stated the non-claims of this monograph at length and gave the reasoning behind each. They are worth restating briefly here, at the point a reader is most likely to be forming an overall impression of what the book has argued, since a summary chapter is exactly where an unstated ambition is most easily read into a text that never asserted it. Structural Semantics does not claim to have shown that any computational system is conscious, that consciousness is unnecessary for understanding, that its formal results settle any question in the philosophy of mind, or that admissibility is a complete or even approximately complete theory of meaning. It does not claim that the resonant reading of activation trajectories in Volume III describes a physical process rather than an analytic one, and it does not claim that the empirical hypotheses proposed there are established rather than proposed. What has been shown, within the scope this monograph set for itself, is narrower and, on its own terms, complete: that admissible structural transformation admits a rigorous mathematical treatment independent of these further

questions, and that this treatment generates concrete, falsifiable predictions when applied to contemporary neural architectures. Everything beyond that remains exactly what Chapter 17 already said it was — open.

18.9 Final Remarks

The principal contribution of this monograph is not the assertion that semantic organization has been completely explained; no single mathematical framework should be expected to resolve every question concerning language, cognition, or intelligence. Instead, the contribution lies in demonstrating that one important aspect of semantic organization can be studied independently, formalized precisely, and connected to existing areas of mathematics through a common structural language. Whether future empirical investigation ultimately confirms or rejects the diagnostic program proposed in Volume III, the mathematical framework developed throughout this work remains available as a language for reasoning about admissible transformations and context-preserving organization. In this sense, Structural Semantics should be viewed less as the end of a research program than as its beginning.

Principal Contributions of This Monograph

For clarity, the principal original contributions of this work are summarized as follows: the formal definition of Structural Semantics as a substrate-independent theory of admissible structural transformations; the Scope Calculus together with its primitive operational alphabet; Bounded Scope Geometry and the Boundary Constraint Tensor; the categorical formulation of admissible transformations through the category Scope; the Representation Invariance Theorem establishing semantic equivalence under admissibility-preserving projections; the interpretation of transformer residual streams as activation trajectories within a dynamical systems framework; the application of analytic signal methods to activation trajectory analysis; a computational diagnostic architecture based on admissibility estimation; and an explicit program of empirical hypotheses, statistical controls, and falsification criteria for evaluating these diagnostic operators.

Closing Statement

Mathematics has long provided languages for describing quantity, symmetry, computation, probability, and geometry. The present work proposes that admissibility deserves recognition as an additional organizing principle through which semantic organization may be investigated. Whether this proposal ultimately becomes a foundational component of future semantic theory or remains a specialized mathematical framework will depend not on philosophical preference, but on the continued interaction of formal proof, computational implementation, and empirical investigation. That interaction has only begun.

Status of Results: Chapter 18

Established background. Summary and synthesis of the mathematical, computational, and empirical developments presented throughout the monograph.

Formal developments. None. This chapter consolidates previously established results rather than introducing new mathematics.

Empirical hypotheses. No new hypotheses are introduced. The chapter summarizes the existing research program and identifies directions for future investigation.

Appendix A

Lambda Calculus and Scope Calculus

The purpose of this appendix is to establish a precise correspondence between the simply typed λ-calculus and the Scope Calculus. Rather than identifying the two systems directly, we construct a translation preserving operational behavior; the principal result is that β-reduction corresponds to a canonical scope transition generated by primitive scope operations. Throughout this appendix we assume familiarity with the simply typed λ-calculus as presented in Barendregt [6] and Pierce [47].

A.1 Translation of Lambda Terms

Definition A.1 (Scope translation)**.** There exists a translation $\mathcal{T} : \Lambda \to \mathrm{Scope}$ defined recursively on the structure of a lambda term by $\mathcal{T}(x) = \mathrm{S}_x$ for a variable x, $\mathcal{T}(\lambda x.M) = \mathcal{B}_d(x, \mathcal{T}(M))$ for an abstraction, and $\mathcal{T}(MN) = \mathcal{P}(\mathcal{T}(M), \mathcal{T}(N))$ for an application.

Definition A.2 (Beta reduction)**.** For lambda terms $(\lambda x.M)N$, the ordinary reduction rule is $(\lambda x.M)N \to_\beta M[x := N]$, substituting N for free occurrences of x in M.

A.2 The Correspondence Theorem

Theorem A.3 (Beta reduction correspondence)**.** *Let $\mathcal{T} : \Lambda \to \mathrm{Scope}$ be the translation of Definition A.1. Every beta reduction $(\lambda x.M)N \to_\beta M[x := N]$ corresponds to an admissible primitive scope transition $\mathcal{P} \circ \mathcal{B}_d \circ \mathcal{C}_l$; equivalently, $\mathcal{T}((\lambda x.M)N)$ reduces, through canonical Scope Calculus reduction, to $\mathcal{T}(M[x := N])$.*

Proof. Application constructs a temporary evaluation scope containing the function object and the argument object, into which the Bind operator introduces a child scope where the formal parameter is locally defined. The Pop operator then transfers control into that child scope, so that Collapse can remove the temporary evaluation frame once substitution is complete. Because canonical reduction preserves admissibility (Theorem 9.3), the resulting scope geometry is isomorphic to the translated substitution term, so beta reduction is represented by the admissible scope transition $\mathcal{P} \circ \mathcal{B}_d \circ \mathcal{C}_l$. □

This correspondence should be read as a mathematical equivalence between two formal systems established through an explicit translation, not as an identification of

syntax: nothing here claims that $\mathcal{P}$, $\mathcal{B}_d$, and $\mathcal{C}_l$ *are* beta reduction, only that beta reduction is faithfully represented by a specific composition of them under $\mathcal{T}$. The remaining appendices build on this translation, progressively situating the Scope Calculus within rewriting theory, category theory, topology, sheaf theory, and measure theory.

Appendix B

The Scope Calculus as an Abstract Reduction System

The purpose of this appendix is to place the Scope Calculus within the well-developed mathematical theory of abstract reduction systems. Throughout the main body of the monograph, reduction has been treated operationally; here we establish the corresponding rewriting theory, proving termination, confluence, and the existence of canonical normal forms under explicit hypotheses, following the general framework of abstract rewriting systems developed by Newman and by Church and Rosser, adapted to the primitive scope algebra.

B.1 Reduction Systems

Definition B.1 (Scope terms)**.** Let $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\} = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$ be the primitive scope alphabet. The set of finite primitive expressions is defined inductively by $t ::= \varepsilon \mid \mathcal{P}(t) \mid \mathcal{R}(t) \mid \mathcal{B}_d(t,t) \mid \mathcal{C}_l(t)$.

Definition B.2 (Reduction relation)**.** A reduction relation $\rightarrow \subseteq \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^* \times \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^*$ is generated by the primitive rewrite rules below. If $t \rightarrow u$, then $C[t] \rightarrow C[u]$ for every admissible context C, extending the relation to arbitrary contexts in the standard manner.

B.2 Primitive Rewrite Rules

The Scope Calculus employs four fundamental rewrite rules, restated here in their fully general term-rewriting form (the elementary word-level rules of Chapter 7 are the special case where each subterm is a variable or ε): identity elimination, $\mathcal{B}_d(x,x) \rightarrow x$; empty collapse, $\mathcal{C}_l(\varepsilon) \rightarrow \varepsilon$; Pop elimination, $\mathcal{P}(\mathcal{B}_d(x,t)) \rightarrow t$; and refusal stability, $\mathcal{R}(\mathcal{R}(t)) \rightarrow \mathcal{R}(t)$. These four reductions generate every higher reduction used throughout the main text.

B.3 Termination

Definition B.3 (Reduction measure)**.** Define $\mu : \Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}^* \rightarrow \mathbb{N}$ by $\mu(\varepsilon) = 0$ and, recursively, $\mu(\mathcal{P}(t)) = 1 + \mu(t)$, $\mu(\mathcal{C}_l(t)) = 1 + \mu(t)$, $\mu(\mathcal{R}(t)) = 1 + \mu(t)$, and $\mu(\mathcal{B}_d(s,t)) = 1 + \mu(s) + \mu(t)$.

Lemma B.4. *Every primitive rewrite strictly decreases μ.*

Proof. Immediate verification for each rewrite rule: each removes at least one constructor while introducing no additional constructors, so $\mu(t') < \mu(t)$ whenever $t \to t'$. □

Theorem B.5 (Strong normalization)**.** *Every reduction sequence terminates.*

Proof. Since μ takes values in $\mathbb{N}$, no infinite strictly decreasing chain can exist, so every reduction sequence is finite. □

B.4 Confluence and Canonical Normal Forms

Strong normalization alone does not guarantee uniqueness of normal forms; we therefore investigate confluence.

Definition B.6. A reduction system is *locally confluent* if $x \to y$ and $x \to z$ imply the existence of w such that $y \to^* w$ and $z \to^* w$.

Lemma B.7. *The primitive Scope Calculus is locally confluent.*

Proof. The proof proceeds by examining every critical overlap between primitive rewrite rules. Since only four primitive rules exist, the overlap set is finite, and each overlap reduces to a common descendant after at most two additional rewrite steps, so every critical pair joins. □

Theorem B.8 (Confluence)**.** *The Scope Calculus is confluent.*

Proof. By the preceding theorem the system is terminating, and by the preceding lemma it is locally confluent; Newman's Lemma therefore implies global confluence. □

Corollary B.9. *Every primitive scope expression possesses a unique normal form.*

Proof. Confluence implies uniqueness of normal forms, and termination guarantees existence. □

B.5 Categorical Interpretation

The reduction system constructed above induces a category whose objects are normal-form scope geometries and whose morphisms are equivalence classes of primitive reduction sequences, with composition induced by concatenation followed by normalization, identity morphisms corresponding to empty reduction sequences, and associativity following from associativity of concatenation. The rewriting system therefore supplies an alternative construction of the category Scope introduced in Chapter 8.

Appendix A established a translation $\mathcal{T} : \Lambda \to \mathrm{Scope}$; the present appendix has shown that $\mathcal{T}$ maps beta reduction into canonical primitive reduction, so the simply typed lambda calculus embeds into the Scope Calculus as a terminating, confluent rewriting subsystem.

B.6 Discussion

The importance of this appendix extends beyond technical completeness. Termination guarantees that scope evaluation always completes; confluence guarantees that evaluation order does not affect the resulting semantic object; and canonical normal forms justify the use of unique semantic representatives throughout Chapters 8–10. Without these results, the Representation Invariance Theorem would depend on evaluation strategy; with them, semantic equivalence becomes an intrinsic property of the Scope Calculus itself. In summary, this appendix has established a formal abstract reduction system for the primitive alphabet, proved strong normalization and local and global confluence, established the existence and uniqueness of canonical normal forms, and reconstructed Scope directly from rewriting theory.

Appendix C

Cartesian Closed Categories and the Scope Calculus

The previous two appendices established complementary descriptions of the Scope Calculus: Appendix A constructed a translation from the simply typed λ-calculus into Scope Geometry, and Appendix B showed that the primitive algebra forms a terminating, confluent abstract reduction system. The objective of the present appendix is to place these constructions within categorical semantics, following Lambek and Scott, by asking under what additional hypotheses Scope can interpret the simply typed lambda calculus as a Cartesian closed category (CCC).

Remark C.1. Every result in this appendix beyond §C.1 is conditional on Scope possessing finite products and exponential objects. Nothing in Chapters 6–9 establishes that these exist for arbitrary bounded scope geometries; the appendix states the hypothesis explicitly wherever it is used, rather than presupposing it.

C.1 Cartesian Closed Structure

Definition C.2 (Terminal object, product, exponential)**.** A category $\mathcal{C}$ has a *terminal object* 1 if for every object A there is a unique morphism $A \to 1$. Objects A, B have a *binary product* $A \times B$, equipped with projections π_1, π_2, if it satisfies the usual universal property; they have an *exponential* B^A if $\mathrm{Hom}(X \times A, B) \cong \mathrm{Hom}(X, B^A)$ naturally in X. A category possessing a terminal object, finite products, and exponentials is *Cartesian closed*.

Theorem C.3 (Conditional Cartesian closure)**.** *If* Scope *admits finite admissible products and exponential scope geometries in the sense above, then* Scope *is Cartesian closed.*

Proof. Immediate from the preceding definition, given the stated hypothesis. □

Within Scope Geometry, an exponential B^A (where it exists) is interpreted as the scope of admissible transformations from A to B, so that functions become first-class scope objects.

C.2 Interpretation of Types and Terms

Under the hypothesis that Scope is Cartesian closed, the simply typed lambda calculus receives a standard categorical interpretation: $[\![A]\!] = A \in \mathrm{Scope}$ for a base type, $[\![A \to$

$B\rrbracket = B^A$ for function types, $[\![A \times B]\!] = A \times B$ for product types, and $[\![1]\!] = 1$ for the unit type, so that every typing judgement becomes a categorical statement.

Lambda abstraction $\lambda x.M$ corresponds to currying: under the translation of Appendix A, $\mathcal{T}(\lambda x.M) = \mathcal{B}_d(x, \mathcal{T}(M))$, and categorically this is represented by the transpose $\Lambda(f) : X \to B^A$, so binding becomes the categorical construction producing an exponential object. Application MN is interpreted through the evaluation morphism $\mathrm{eval} : B^A \times A \to B$; within the Scope Calculus, evaluation corresponds to the admissible sequence $\mathcal{P} \circ \mathcal{C}_l$, with the evaluation scope entered, the bound environment activated, and the temporary computation collapsed after completion.

Theorem C.4 (Categorical beta correspondence)**.** *Under the Cartesian closure hypothesis, beta reduction $(\lambda x.M)N \to_\beta M[x := N]$ commutes with the translation $\mathcal{T}$: applying evaluation to the curried translation of $\lambda x.M$ and $\mathcal{T}(N)$ and reducing to canonical form yields $\mathcal{T}(M[x := N])$.*

Proof. Application corresponds to evaluation and lambda abstraction to exponential transpose; the universal property of exponentials guarantees that evaluation following currying commutes. Since primitive reduction is confluent (Appendix B), normalization is unique, so the translated computation reduces uniquely to the translated substituted term. □

C.3 Curry–Howard–Lambek Interpretation

The categorical semantics, when it applies, induces a logical interpretation in which types correspond to propositions, morphisms to proofs, scope transformations to proof normalization, and primitive reductions to proof simplification: canonical scope reduction is simultaneously program evaluation, proof normalization, categorical composition, and scope simplification, placing the Scope Calculus within the broader Curry–Howard–Lambek correspondence.

The Representation Invariance Theorem (Theorem 9.10) acquires a categorical reading under this interpretation: different computational substrates correspond to different concrete categories, and projection into Scope forgets implementation-specific structure while preserving admissible composition, so representation invariance becomes, in this reading, a statement about commuting categorical diagrams rather than about coordinate representations.

Conjecture C.5 (Universal property of Scope Geometry)**.** Among categories generated by admissible computational transitions, Scope is initial with respect to primitive scope-preserving functors.

This is stated as a conjecture, not a theorem: nothing established in this monograph proves it, and it is included to indicate a direction for future work rather than a result relied upon elsewhere in the text. If established, every admissibility-preserving computational system would factor uniquely through Scope.

C.4 Discussion

The value of the Cartesian closed reading is not mathematical elegance for its own sake, but compatibility: under the stated hypothesis, the Scope Calculus connects to the standard semantic foundations of functional programming, typed lambda calculus, proof

theory, and categorical logic, rather than replacing them. Scope Geometry extends these theories by making admissibility an explicit primitive mathematical object, unifying functional evaluation, scope management, proof normalization, and admissible computation within a common categorical language. This appendix has established the conditional Cartesian closed interpretation of Scope, the interpretation of lambda abstraction as binding and application as admissible evaluation, a strengthened categorical proof of the beta correspondence, the Curry–Howard–Lambek reading, and an explicitly labeled open conjecture concerning a universal property of Scope Geometry.

Appendix D

Higher Categories, Double Categories, and Scope Geometry

Appendix C showed that Scope admits a Cartesian closed interpretation under appropriate hypotheses, with objects representing bounded scope geometries and morphisms representing admissible transformations. Many computational systems, however, exhibit two independent notions of composition simultaneously: a compiler performs both control-flow and data-flow composition, a proof assistant distinguishes proof refinement from proof composition, and Scope Geometry itself naturally possesses both admissible computational transitions and refinement of scope description. This observation motivates the introduction of higher categorical structure, developed here as a conditional extension rather than as an established part of the core framework.

D.1 Double Categories

Definition D.1 (Double category)**.** A *double category* $\mathbb{D}$ consists of objects, horizontal morphisms, vertical morphisms, and squares (2-cells), together with compatible horizontal and vertical composition satisfying the interchange law. Unlike an ordinary category, a double category allows two independent notions of composition to coexist.

Definition D.2 (Scope double category, conditional)**.** Suppose Scope admits a well-defined refinement relation on scope geometries, compatible with admissible transitions in the sense made precise below. Under this hypothesis, define $\mathbb{Scope}$ by taking objects to be scope geometries, horizontal morphisms to be admissible computational transitions, vertical morphisms to be scope refinement operations, and squares to be admissible commuting refinements.

In this construction, the primitive scope operators become generators for the higher-dimensional cells: $\mathcal{P}$ corresponds to entering a neighboring scope, $\mathcal{B}_d$ to constructing a new refinement layer, $\mathcal{C}_l$ to evaluation of a completed square, and $\mathcal{R}$ to a degenerate square whose refinement is forbidden. Scope computation, read this way, is no longer merely sequential but possesses internal geometric structure. Horizontal composition, for $f : \mathrm{S}_1 \to \mathrm{S}_2$ and $g : \mathrm{S}_2 \to \mathrm{S}_3$, coincides with the ordinary composition of Chapter 8; vertical composition, for refinements r_1, r_2, corresponds to successive refinement of admissibility constraints and changes descriptive resolution rather than execution order.

Theorem D.3 (Interchange law, conditional)**.** *For every compatible square in a well-defined* $\mathbb{Scope}$, $(\alpha \circ_v \beta) \circ_h (\gamma \circ_v \delta) = (\alpha \circ_h \gamma) \circ_v (\beta \circ_h \delta)$.

Proof. Both compositions produce identical refinement histories, since horizontal computation preserves admissibility while vertical refinement preserves scope inclusion, so the resulting square commutes. □

The interchange law guarantees that, under the stated hypotheses, refinement order and execution order remain compatible. A functor $\mathcal{R} : \mathrm{Scope} \to \mathrm{Scope}$ that refines every scope by introducing additional internal structure while preserving identities, composition, and admissibility models progressively more detailed semantic analysis without altering external computational behavior; repeated refinement then produces a hierarchical tower $\mathrm{S}^{(0)} \to \mathrm{S}^{(1)} \to \mathrm{S}^{(2)} \to \cdots$ of increasingly refined semantic descriptions, each preserving admissibility while increasing structural detail.

D.2 Bicategorical Relaxation

Many computational transformations preserve admissibility only up to canonical equivalence, so strict equality may be unnecessarily restrictive.

Definition D.4 (Bicategorical Scope Geometry, conditional)**.** A bicategorical Scope Geometry replaces equalities $f \circ g = h$ by coherent natural isomorphisms $f \circ g \cong h$.

This relaxation accommodates optimization, parallel evaluation, lazy computation, and compiler transformations, all of which preserve semantics without preserving identical execution histories.

Theorem D.5 (Coherence, conditional)**.** *If primitive reductions are confluent, every admissible bicategorical diagram generated by primitive scope operations commutes after normalization.*

Proof. Appendix B established unique normal forms, so every composite reduces uniquely, and hence every coherence diagram commutes after normalization. □

D.3 Computational Interpretation

Within practical programming systems, horizontal morphisms correspond to execution, vertical morphisms to abstraction refinement, squares to verified implementation, and higher cells to proofs that independently constructed implementations preserve admissibility. Higher Scope Geometry, read this way, provides a candidate mathematical language for verified compilers, proof assistants, interactive theorem provers, incremental program synthesis, and hierarchical autonomous agents — though, as throughout this appendix, this reading is offered as a direction rather than as an established result.

Conjecture D.6 (Toward higher scope categories)**.** The constructions above generalize to (∞, n) scope categories, whose higher morphisms would encode increasingly sophisticated notions of semantic equivalence, proof refinement, program transformation, and admissibility preservation, potentially connecting to homotopy type theory.

D.4 Discussion

The progression from ordinary categories through Cartesian closed categories to double categories, bicategories, and conjecturally to higher categories illustrates a theme of this

monograph: Scope Geometry is not merely an operational formalism, but appears compatible with progressively richer categorical structures while preserving its primitive operational interpretation, provided the relevant hypotheses (refinement relations, admissible products and exponentials, confluent bicategorical diagrams) are made explicit rather than assumed. This appendix has developed the double-categorical formulation of Scope Geometry conditionally on the existence of a compatible refinement relation, identified the primitive operators as generators of higher-dimensional cells, proved the interchange law and a coherence theorem under their respective stated hypotheses, and outlined hierarchical refinement towers and a conjectural extension toward higher category theory.

Appendix E

Topological Semantics of Scope Geometry

The preceding appendices developed the Scope Calculus from the perspectives of rewriting systems and category theory. We now introduce a topological formulation, showing that admissibility admits a natural interpretation in terms of continuity, closure, interiors, boundaries, and connected components. Where the earlier appendices emphasized algebraic computation, the present one studies the geometric organization of semantic spaces, assuming ordinary point-set topology as developed by Munkres [40].

E.1 Scope Spaces

Definition E.1 (Scope topological space)**.** A *scope space* is a pair (S, τ) where S is the underlying set of semantic configurations and τ is a topology on it satisfying the usual axioms.

The topology describes semantic neighborhoods rather than geometric distance: nearby points correspond to semantically similar admissible configurations.

Definition E.2 (Interior, closure, boundary)**.** The *interior* operator is $\mathrm{Int}(A) = \bigcup\{U \in \tau : U \subseteq A\}$; the *closure* operator is $\mathrm{Cl}(A) = \bigcap\{F : F \text{ closed}, A \subseteq F\}$; and the *boundary* of A is $\partial A = \mathrm{Cl}(A) \setminus \mathrm{Int}(A)$.

Interior points correspond to configurations whose admissibility remains stable under sufficiently small perturbations — configurations lying safely within a semantic region. Boundary points separate admissible from inadmissible semantic regions, so the boundary constraint of Chapter 6 admits a topological reading as an operator governing movement across ∂A. Closure represents completion of semantic neighborhoods: configurations lying in the closure may be approached arbitrarily closely through admissible computation.

E.2 Continuity and Admissible Maps

Definition E.3 (Continuity)**.** A scope transformation $f : (\mathrm{S}, \tau_1) \to (\mathrm{S}, \tau_2)$ is *continuous* if $f^{-1}(U)$ is open whenever U is open.

Continuity expresses preservation of semantic neighborhoods: small admissible perturbations remain small after transformation. The principal topological object of Structural Semantics is stronger than mere continuity.

Definition E.4 (Admissible map)**.** A continuous map $f : \mathrm{S} \to \mathrm{S}$ is *admissible* if $f(\mathrm{Int}(A)) \subseteq \mathrm{Int}(f(A))$ and $f(\partial A) \subseteq \partial(f(A))$.

Admissible maps therefore preserve both semantic interiors and semantic boundaries. Two configurations x, y lie in the same *admissible component* if they belong to a common connected subset of S; connected components represent maximal regions through which continuous admissible computation may proceed, and moving between disconnected components necessarily requires violating some admissibility constraint. A semantic region $K \subseteq \mathrm{S}$ is *compact* if every open cover possesses a finite subcover; compact regions admit useful properties for this theory, including that continuous admissible functions attain extrema on them and that optimization over admissible states becomes well behaved.

E.3 Fixed Points and Homotopy

Topological fixed-point theory connects naturally with the repair operators introduced in the reconstruction chapters.

Theorem E.5 (Topological repair principle)**.** *Let K be a compact convex admissible region and $R : K \to K$ a continuous repair operator. Then R possesses at least one fixed point.*

Proof. Immediate from Brouwer's Fixed Point Theorem. □

This theorem provides a classical topological justification for the existence of repaired admissible configurations. Two admissible maps $f, g : \mathrm{S} \to \mathrm{S}$ are *homotopic* if there is a continuous $H : \mathrm{S} \times [0, 1] \to \mathrm{S}$ with $H(x, 0) = f(x)$ and $H(x, 1) = g(x)$, so that homotopy represents continuous deformation of admissible computations. Loops in semantic space measure global computational structure through the fundamental group $\pi_1(\mathrm{S})$, the homotopy classes of admissible loops based at a chosen configuration; a non-trivial fundamental group indicates a semantic obstacle that cannot be removed through continuous admissible deformation. The algebraic-topological invariants π_n, homology, cohomology, and Euler characteristic remain unchanged under admissible homotopy equivalence, and so provide candidate semantic invariants independent of any particular representation, suggesting a future connection with persistent homology and topological data analysis.

E.4 Discussion

Topology supplies an additional geometric interpretation of the Scope Calculus: primitive operations become continuous movements between semantic regions, admissibility becomes boundary preservation, repair corresponds to continuous deformation toward fixed points, and representation invariance becomes topological equivalence rather than coordinate equality. This perspective complements, rather than replaces, the categorical interpretation of the preceding appendices. In summary, this appendix has introduced scope topological spaces, given interior, closure, and boundary readings of admissibility, defined continuous and admissible maps, established connected components and compactness for admissible regions, proved a topological repair theorem via Brouwer's theorem, and introduced the homotopy interpretation of semantic evolution together with the fundamental group as a candidate semantic invariant.

Appendix F

Sheaf Theory and Local-to-Global Reconstruction

The preceding appendix gave a topological interpretation of Scope Geometry, in which semantic configurations were points of a topological space and admissibility was boundary preservation. Topology alone, however, cannot fully describe semantic organization: many computational systems possess information that is locally consistent while globally inconsistent, and conversely global coherence frequently emerges only through reconciling multiple local observations. The mathematical language designed for this problem is sheaf theory, originally developed in algebraic topology and algebraic geometry and now applied throughout differential geometry, distributed systems, and logic. Within Structural Semantics, sheaf theory provides the mathematical language of reconstruction.

F.1 Presheaves and Sheaves

Let (S, τ) be a scope topological space and let $\mathcal{U} = \{U_i\}_{i \in I}$ be an open cover, $\mathrm{S} = \bigcup_i U_i$, each U_i representing a locally observable semantic region; no single region need contain sufficient information to reconstruct the entire computational state.

Definition F.1 (Scope presheaf)**.** A *scope presheaf* $\mathcal{F}$ assigns a set $\mathcal{F}(U)$ to every open $U \subseteq \mathrm{S}$, together with restriction maps $\rho_{UV} : \mathcal{F}(U) \to \mathcal{F}(V)$ for $V \subseteq U$, satisfying $\rho_{UU} = \mathrm{id}$ and $\rho_{VW} \circ \rho_{UV} = \rho_{UW}$.

Intuitively $\mathcal{F}(U)$ contains every semantic description observable within U. An element $s \in \mathcal{F}(U)$ is a *local section*, representing a partial semantic description extracted from a restricted computational context — a local variable environment, an attention neighborhood, a proof fragment, or a segment of an activation trajectory. Restriction $\rho_{UV}(s)$ for $V \subseteq U$ models information loss under localization.

Definition F.2 (Scope sheaf)**.** A presheaf $\mathcal{F}$ is a *sheaf* if it satisfies local identity (if $s, t \in \mathcal{F}(U)$ agree on every member of an open cover of U, then $s = t$) and gluing (if sections $s_i \in \mathcal{F}(U_i)$ agree on overlaps, $\rho_{U_i, U_i \cap U_j}(s_i) = \rho_{U_j, U_i \cap U_j}(s_j)$, then there is a unique $s \in \mathcal{F}(U)$ restricting to every s_i).

F.2 Reconstruction as Gluing

The gluing axiom has a direct computational reading: compatible local semantic descriptions reconstruct a unique global semantic configuration, a principle underlying distributed reasoning, proof composition, compiler passes, modular verification, and hierarchical language understanding.

Definition F.3 (Reconstruction operator)**.** Let $\mathcal{U} = \{U_i\}$ be an admissible cover. Define $\mathcal{R} : \prod_i \mathcal{F}(U_i) \to \mathcal{F}(\mathrm{S})$ by $\mathcal{R}(s_1, \ldots, s_n) = \mathrm{Glue}(s_1, \ldots, s_n)$ whenever the sections agree on overlaps.

Reconstruction, in this reading, is precisely the sheaf gluing operator. Not every family of local observations admits gluing: the *reconstruction defect* is 0 whenever gluing succeeds uniquely and positive otherwise, giving a sheaf-theoretic reading of the reconstruction defect discussed in Chapter 9 as a measure of failure of global compatibility rather than of numerical error alone. The *stalk* at a configuration x, $\mathcal{F}_x = \varinjlim_{x \in U} \mathcal{F}(U)$, collects every local semantic observation arbitrarily close to x; a scope-preserving transformation $f : \mathrm{S} \to \mathrm{S}$ induces a sheaf morphism that preserves restriction maps, commutes with gluing, and therefore preserves admissible reconstruction.

F.3 Cohomological Obstructions

When reconstruction fails, the obstruction can itself be measured. Given an admissible open cover, the associated Čech complex computes compatibility between overlapping local descriptions, and the resulting cohomology groups $\check{H}^k(\mathrm{S}, \mathcal{F})$ measure increasingly global reconstruction obstructions; in particular a nontrivial $\check{H}^1$ indicates that locally compatible semantic fragments cannot be assembled into a globally admissible configuration.

Conjecture F.4 (Cohomological admissibility)**.** A computational system is globally admissible whenever the associated reconstruction cohomology vanishes: $\check{H}^1(\mathrm{S}, \mathcal{F}) = 0$.

This is stated as a conjecture rather than a theorem; it generalizes the reconstruction results of Chapter 9 and suggests a connection between admissibility and algebraic topology that is not established here.

F.4 Discussion

Sheaf theory provides one of the strongest mathematical interpretations of Structural Semantics developed in this monograph: Scope Geometry describes admissible regions, topology describes continuity, category theory describes composition, and sheaf theory explains how local admissibility reconstructs global semantic organization. Read this way, reconstruction is not a heuristic procedure but an instance of one of the central local-to-global principles of modern mathematics. This appendix has introduced scope presheaves and sheaves, local sections and restriction morphisms, sheaf gluing as a reading of semantic reconstruction, the reconstruction defect as gluing failure, stalk semantics, sheaf morphisms, and a Čech-cohomological reading of reconstruction failure together with the associated open conjecture.

Appendix G

Measure Theory, Entropy, and Admissibility

The preceding appendices developed the Scope Calculus through rewriting theory, category theory, topology, and sheaf theory, each providing an increasingly abstract structural language for admissible computation. The present appendix instead introduces quantitative measures on semantic spaces, drawing on measure theory, probability, and information theory, so that admissibility becomes not merely a binary property but a measurable quantity. Throughout, $(\mathrm{S}, \Sigma, \mu)$ denotes a measurable scope space.

G.1 Measures and Admissibility

Definition G.1 (Scope σ-algebra and semantic measure)**.** A *scope σ-algebra* Σ is a collection of subsets of S containing $\emptyset$ and closed under complements and countable unions; its elements are *measurable semantic regions*. A *semantic measure* $\mu : \Sigma \to [0, \infty]$ satisfies $\mu(\emptyset) = 0$ and countable additivity over disjoint measurable sets. Unlike a probability measure, a semantic measure need not satisfy $\mu(\mathrm{S}) = 1$; normalizing μ by $\mu(\mathrm{S})$ recovers a probability measure as one special instance of semantic measurement.

Definition G.2 (Admissibility measure)**.** Let $A \subseteq \mathrm{S}$ with $\mu(A) > 0$. Its *admissibility* is $\alpha(A) = \mu(A \cap \mathsf{Adm})/\mu(A)$, so that $\alpha = 1$ indicates complete admissibility and $\alpha = 0$ indicates complete inadmissibility.

For a stochastic process $X : \Omega \to \mathrm{S}$, the *expected admissibility* $E[\alpha] = \int \alpha(X)\, dP$ measures the average structural integrity of the process.

G.2 Entropy and Mutual Information

For a probability distribution $P = (p_i)$, the Shannon entropy is $H(P) = -\sum_i p_i \log p_i$; within Structural Semantics, entropy measures uncertainty over admissible semantic configurations rather than merely over tokens. For a semantic state X and an observation Y, the conditional entropy $H(X \mid Y)$ measures remaining uncertainty after observation, and the mutual information $I(X;Y) = H(X) - H(X \mid Y)$ measures preserved structure: large mutual information indicates that observation preserves semantic organization, small mutual information suggests context loss. For an admissible distribution P and an observed distribution Q, the relative entropy $D_{\mathrm{KL}}(P \| Q) = \sum_i p_i \log(p_i/q_i)$ measures deviation from admissible behavior.

Definition G.3 (Reconstruction entropy)**.** Let R be the reconstruction operator of Appendix F. Define the reconstruction entropy $H_R = H(R(X))$.

Theorem G.4 (Entropy monotonicity)**.** *If R is an admissibility-preserving reconstruction operator, then $H(R(X)) \leq H(X)$.*

Proof. Under admissibility, reconstruction identifies compatible semantic regions without creating new ambiguity, so the image partition refines rather than enlarges the underlying uncertainty, and entropy cannot increase. □

Observation updates admissibility in the usual Bayesian manner: for new evidence E, $P(A \mid E) = P(E \mid A)P(A)/P(E)$, so the posterior updates admissibility estimates while preserving measure-theoretic consistency.

G.3 Measure-Preserving Dynamics

A scope transformation $T : \mathrm{S} \to \mathrm{S}$ is *measure preserving* if $\mu(T^{-1}(A)) = \mu(A)$ for every measurable A, so that measure-preserving admissible transformations conserve semantic volume. Such a transformation is *ergodic* if every invariant measurable subset has measure 0 or 1, connecting long-running computation with statistical semantic equilibrium.

G.4 Discussion

This appendix extends Structural Semantics from qualitative to quantitative analysis: topology supplied continuity, category theory supplied composition, sheaf theory supplied reconstruction, and measure theory now supplies uncertainty, probability, entropy, and statistical inference, so that admissibility can be studied simultaneously from geometric, algebraic, topological, and statistical perspectives. This appendix has introduced measurable scope spaces and semantic measures, defined the admissibility measure and expected admissibility, related entropy, conditional entropy, and mutual information to semantic uncertainty and its preservation, proved an entropy-monotonicity theorem for admissibility-preserving reconstruction, and introduced measure-preserving and ergodic scope dynamics.

Appendix H

Notation and Mathematical Conventions

This appendix collects, in one place, every symbol, operator, and convention used throughout the monograph, as promised in the front-matter Notation and Conventions section. Symbols are grouped by the cluster of chapters in which they are introduced; within each group they appear in the order first used.

H.1 Scope Calculus (Chapters 1, 6–10)

The category of scope geometries is Scope; a scope object is written $\mathrm{S} = \langle \mathcal{B}, \mathcal{E}, \mathcal{S}_{\text{child}} \rangle$, with $\mathcal{B}$ the boundary (a collection of admissibility predicates), $\mathcal{E}$ the admissible interior, and $\mathcal{S}_{\text{child}}$ the collection of child scopes. A scope morphism is $\sigma : \mathrm{S}_i \to \mathrm{S}_j$. The primitive alphabet is $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\} = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$, where $\mathcal{P}$ promotes an entity from a child scope to its parent, $\mathcal{R}$ rejects an inadmissible transition, $\mathcal{B}_d$ creates an admissible dependency within a scope, and $\mathcal{C}_l$ resolves a scope into a canonical result. A computational system is $M = (\mathcal{D}, \mathcal{T})$, with $\mathcal{D}$ a state domain and $\mathcal{T}$ the set of admissible transition operators $\tau : \mathcal{D} \to \mathcal{D}$. The category of Whiteheadian processes, used only in Chapters 4, 5, and 10, is $\mathrm{Process}$; the translation functor between the two is $F : \mathrm{Process} \to \mathrm{Scope}$.

H.2 Whitehead Reconstruction (Chapters 4–5, 10)

An actual occasion is E; antecedent data is D; a prehension is $P = \langle D, v, A \rangle$, with $v \in \{+, -\}$ the valuation polarity and A the Subjective Form; the satisfaction state is S. These symbols are kept visually distinct from the Scope Calculus notation throughout, since no identification between the two systems is assumed before the translation functor of Chapter 10 makes any correspondence explicit.

H.3 Transformer Analysis (Chapters 11–15)

The residual stream tensor is $\mathrm{X} \in \mathbb{R}^{(L+1) \times T \times d_{\text{model}}}$, with L the number of layers, T the sequence length, and d_{model} the embedding dimension; $\mathrm{x}_{l,t}$ is the residual vector for token t after layer l. An activation trajectory is $\gamma_t = (\mathrm{x}_{0,t}, \ldots, \mathrm{x}_{L,t})$; the layerwise Jacobian is $J_l = \partial \mathrm{x}_{l+1} / \partial \mathrm{x}_l$; discrete curvature is κ_l; the local layerwise Lyapunov exponent is $\lambda_l = \log \sigma_{\max}(J_l)$. The analytic activation signal is $\mathrm{z}_l = \mathrm{x}_l + i\, \mathcal{H}(\mathrm{x}_l)$, with $\mathcal{H}$ the discrete

Hilbert transform; instantaneous amplitude and phase are A_l and ϕ_l; instantaneous frequency is $\omega_l = \phi_{l+1} - \phi_l$; the Phase Locking Value is PLV $\in [0, 1]$; the Marine Operator is $\mathcal{M} : \Gamma \to \mathbb{R}^k$ on the space Γ of activation trajectories. The Doppler velocity is v_l and the Marine salience gate is G_l. The diagnostic state vector is $\mathrm{d}_t = (A, \phi, \omega, \mathrm{PLV}, \kappa, \lambda)$; the admissibility estimator is $\mathcal{A} : \mathrm{d} \to [0, 1]$; the context collapse threshold is $\tau \in (0, 1)$.

H.4 General Conventions

Bold uppercase symbols denote categories or structured spaces; calligraphic symbols denote operators, constraint systems, or families of sets; lowercase Greek letters denote morphisms, transformations, or parameters. Once a symbol is assigned a formal meaning in a given cluster of chapters, it is preserved throughout the monograph and is not reassigned. Definitions, propositions, lemmas, theorems, and corollaries share a single numbering sequence within each chapter, restarting at each chapter; equations are numbered within chapters. Every theorem's hypotheses are stated explicitly in its statement rather than left to surrounding prose.

Appendix I

Collected Proofs

This appendix collects the proofs deferred from the main text for length, principally the structural-induction argument underlying Operational Completeness (Chapter 7) and the closure and decomposition arguments underlying the foundational theorems of Chapter 9. Each proof below restates its theorem for reference.

I.1 Operational Completeness (Theorem 7.6)

Theorem. Every admissible transformation of bounded scope geometries admits a finite presentation as a composition of primitive operations from $\Sigma = \{\mathcal{P}, \mathcal{R}, \mathcal{B}_d, \mathcal{C}_l\}$.

Proof. We argue by induction on the number of atomic modifications comprising an admissible transformation, where an atomic modification changes exactly one of $\mathcal{B}$, $\mathcal{E}$, or $\mathcal{S}_{\text{child}}$ at a single scope.

Base case. A transformation comprising zero atomic modifications is the identity, presented by the empty primitive sequence.

Inductive step. Suppose every admissible transformation comprising n atomic modifications admits a finite primitive presentation. Let T comprise $n+1$ atomic modifications, and let T' be T with its final atomic modification removed, so T' comprises n modifications and, by the inductive hypothesis, has a finite primitive presentation w'. The final modification either introduces new admissible information into $\mathcal{E}$ (representable by $\mathcal{B}_d$, if it creates a dependency, or by an update already captured within a preceding $\mathcal{B}_d$ or $\mathcal{P}$ otherwise), rejects an inadmissible candidate transition (representable by $\mathcal{R}$), transfers information from a child scope into its parent (representable by $\mathcal{P}$), or resolves the scope into a canonical result (representable by $\mathcal{C}_l$); one of these four cases must apply because Chapter 6 characterized every admissibility-relevant change to a scope geometry as falling into exactly one of them. Appending the corresponding primitive to w' yields a finite primitive presentation of T. By induction, every admissible transformation with finitely many atomic modifications admits a finite primitive presentation, and every admissible transformation has finitely many atomic modifications by the finiteness assumption of Chapter 9. □

I.2 Local Confluence Verification (Chapter 7, §7 and Appendix B)

The elementary rewrite rules (R1)–(R4) of Chapter 7 give rise to a finite set of critical pairs, since each rule's left-hand side has bounded depth and there are only four rules. Enumerating the overlaps: (R1) (identity elimination) can overlap with any other rule

only at the empty word, where all rules agree trivially since ε has no further structure to rewrite. (R2) and (R3) (consecutive refusal and consecutive collapse) overlap only with themselves, and each such overlap ($\mathcal{R}\mathcal{R}\mathcal{R}$, $\mathcal{C}_l\mathcal{C}_l\mathcal{C}_l$) reduces to the single-operator form by two applications of the same rule regardless of order, so both branches join immediately. (R4) (empty promotion) overlaps with (R1) only at $\mathcal{P}\varepsilon$ itself, where both rules yield ε directly. No other overlaps exist among (R1)–(R4), since each rule's left-hand side pattern is disjoint from the others outside the cases just enumerated. Hence every critical pair joins within at most two steps, establishing local confluence as asserted in Chapter 7 and used in Appendix B.

I.3 Closure, Decomposition, and Reconstruction (Chapter 9)

The Closure Theorem (Theorem 9.1) follows by induction on the primitive presentation of σ furnished by Proposition 8.4: each of $\mathcal{P}$, $\mathcal{R}$, $\mathcal{B}_d$, and $\mathcal{C}_l$ was defined in Chapter 7 to act only on admissible states and to produce a well-formed bounded scope geometry satisfying Definition 6.5 as output (boundary inheritance is preserved by construction, since none of the four primitives removes an inherited constraint), so a finite composition of them, applied to a well-formed input, returns a well-formed output.

The Scope Decomposition Theorem (Theorem 9.2) is Proposition 8.4 restated at the level of Scope; the induction is the one given in §I7.6 above, transported along the identification of morphisms with admissible transformations.

The State Reconstruction Theorem depends on collapse operations recording their boundary history; under that recording assumption, the parent scope S_p at any step is determined as the unique bounded scope geometry whose interior and boundary agree, on every child region present at that step, with the recorded pre-collapse data — uniqueness follows because two candidate parents agreeing with all recorded child data on their boundaries would, by the sheaf-theoretic local-identity property of Appendix F applied to the cover by child scopes, coincide.

Bibliography

[1] Shun-ichi Amari and Hiroshi Nagaoka. *Methods of Information Geometry*. American Mathematical Society, 2016.

[2] Anthropic. Towards monosemanticity: Decomposing language models with dictionary learning. Transformer Circuits Thread, 2023.

[3] W. Ross Ashby. *An Introduction to Cybernetics*. Chapman & Hall, 1956.

[4] Steve Awodey. *Category Theory*. Oxford University Press, 2010.

[5] Franz Baader and Tobias Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

[6] Henk P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North-Holland, 1984.

[7] Ned Block. On a confusion about a function of consciousness. *Behavioral and Brain Sciences*, 18(2):227–247, 1995.

[8] Tom B. Brown et al. Language models are few-shot learners. arXiv:2005.14165, 2020.

[9] Stanley Burris and H. P. Sankappanavar. *A Course in Universal Algebra*. Springer, 1981.

[10] David J. Chalmers. *The Conscious Mind: In Search of a Fundamental Theory*. Oxford University Press, 1996.

[11] Andy Clark. *Whatever Next? Predictive Brains, Situated Agents, and the Future of Cognitive Science*, volume 36. 2013.

[12] Eugenio Coseriu. *Principios de Semántica Estructural*. Gredos, 1977.

[13] Ferdinand de Saussure. *Cours de Linguistique Générale*. Payot, 1916.

[14] Samuel Eilenberg and Saunders Mac Lane. General theory of natural equivalences. *Transactions of the American Mathematical Society*, 58:231–294, 1945.

[15] Nelson Elhage et al. A mathematical framework for transformer circuits. Anthropic, 2021.

[16] Jerry A. Fodor. The language of thought. 1975.

[17] Gottlob Frege. Über sinn und bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, 100:25–50, 1892.

[18] Karl Friston. The free-energy principle: A unified brain theory? *Nature Reviews Neuroscience*, 11:127–138, 2010.

[19] Karl Friston. A free energy principle for a particular physics. arXiv:1906.10184, 2019.

[20] Gerhard Gentzen. Untersuchungen über das logische schließen. Mathematische Zeitschrift, 1935.

[21] Mor Geva et al. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. arXiv:2203.14680, 2022.

[22] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[23] Algirdas Julien Greimas. *Sémantique Structurale*. Larousse, 1966.

[24] Martin Heidegger. *Being and Time*. Niemeyer, 1927.

[25] Louis Hjelmslev. *Omkring Sprogteoriens Grundlæggelse*. Munksgaard, 1943.

[26] Louis Hjelmslev. *Prolegomena to a Theory of Language*. University of Wisconsin Press, 1961.

[27] Jordan Hoffmann et al. Training compute-optimal large language models. arXiv:2203.15556, 2022.

[28] Jakob Hohwy. *The Predictive Mind*. Oxford University Press, 2013.

[29] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 2006.

[30] Edmund Husserl. *Ideas: General Introduction to Pure Phenomenology*. Niemeyer, 1913.

[31] Jenann Ismael. Curie's principle. *Synthese*, 110(2):167–190, 1997.

[32] Jenann Ismael and Bas C. van Fraassen. Symmetry as a guide to superfluous theoretical structure. In Katherine Brading and Elena Castellani, editors, *Symmetries in Physics: Philosophical Reflections*, pages 371–392. Cambridge University Press, 2003.

[33] Jared Kaplan et al. Scaling laws for neural language models. arXiv:2001.08361, 2020.

[34] Cheng Luo, Zefan Cai, and Junjie Hu. Multi-head attention residuals. arXiv:2607.27230, 2026.

[35] John Lyons. *Structural Semantics*. Blackwell, 1963.

[36] John Lyons. *Semantics*. Cambridge University Press, 1977.

[37] Saunders Mac Lane. *Categories for the Working Mathematician*. Springer, 1998.

[38] Samuel Marks et al. Sparse feature circuits. arXiv, 2024.

[39] Maurice Merleau-Ponty. *Phenomenology of Perception*. Gallimard, 1945.

[40] James R. Munkres. *Topology*. Prentice Hall, 2000.

[41] Thomas Nagel. What is it like to be a bat? *The Philosophical Review*, 83(4):435–450, 1974.

[42] Neel Nanda et al. Progress measures for grokking via mechanistic interpretability. arXiv:2301.05217, 2023.

[43] Allen Newell and Herbert A. Simon. *Computer Science as Empirical Inquiry: Symbols and Search*, volume 19. 1976.

[44] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. Distill, 2020.

[45] OpenAI. Gpt-4 technical report. arXiv:2303.08774, 2023.

[46] Charles Sanders Peirce. *Collected Papers of Charles Sanders Peirce*. Harvard University Press, 1931.

[47] Benjamin C. Pierce. *Types and Programming Languages*. MIT Press, 2002.

[48] Hilary Putnam. Psychological predicates. *Art, Mind, and Religion*, pages 37–48, 1967.

[49] Dana Scott. Outline of a mathematical theory of computation. *Technical Monograph PRG-2, Oxford University Computing Laboratory*, 1970.

[50] John R. Searle. Minds, brains, and programs. *Behavioral and Brain Sciences*, 3(3): 417–457, 1980.

[51] Thomas L. Short. *Peirce's Theory of Signs*. Cambridge University Press, 2007.

[52] Adly Templeton et al. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. Transformer Circuits Thread, 2024.

[53] Jost Trier. Der deutsche wortschatz im sinnbezirk des verstandes. *Universitätsverlag Winter*, 1931.

[54] Alan M. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.

[55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[56] Alfred North Whitehead. *Process and Reality*. Macmillan, 1929.

[57] Norbert Wiener. *Cybernetics: Or Control and Communication in the Animal and the Machine*. MIT Press, 1948.

[58] Glynn Winskel. *The Formal Semantics of Programming Languages*. MIT Press, 1993.

Index of Symbols