Tokenization + Algebra: How Language Models Learn to Count, Classify, and Compute

By Francisco Revelles

Introduction

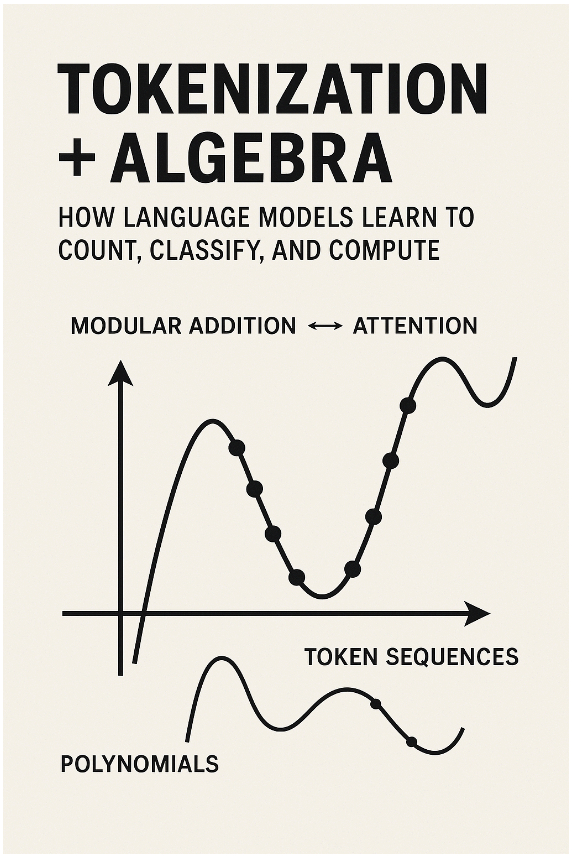
TOKENIZATION + ALGEBRA
HOW LANGUAGE MODELS LEARN TO COUNT, CLASSIFY, AND COMPUTE
MODULAR ADDITION ⟷ ATTENTION
TOKEN SEQUENCES
POLYNOMIALS

Figure 0 — Cover illustration for Tokenization + Algebra.
The graphic highlights the central parallels between algebra and language models: polynomials with token sequences, and modular addition with attention.

Language Models (LLMs) aren't just statistical parrots—they're algebraic machines. Every token they process behaves like a number, and the operations inside these models resemble the same arithmetic principles that underlie cryptography, error correction, and number theory.

In this article, I argue that **tokenization is not just a preprocessing trick but a gateway to algebra**. From token IDs to embeddings, from attention matrices to emergent reasoning, the structures inside LLMs can be seen through the lens of modular arithmetic.

. . .

1. Tokenization as a Gateway to Algebra

Tokenization maps language into a finite alphabet of symbols, much like integers modulo (*p*).

- **Tokens are atomic**, each with a unique integer ID.
- **Token embeddings are vectors** in continuous space, but they transmit information in ways that act modular or categorical.
- **Sequences of tokens resemble polynomials** over finite fields, creating a natural analogy for reasoning about LLMs.

Seen this way, the building blocks of language processing are not arbitrary symbols—they're algebraic structures in disguise.

. . .

2. Modular Arithmetic Inside LLMs

As I explored in an earlier essay (*How Modular Arithmetic Powers Everything from AI to Cryptography*), LLMs don't just represent tokens: they **operate on them algebraically**.

- Hidden layers encode sequences in ways that resemble **modular addition and multiplication**.
- Embedding spaces and attention matrices exhibit **periodic patterns**, mirroring cyclic groups in algebra.

In transformers, wrapping around modular boundaries is analogous to "rollover" operations in arithmetic. What looks like vector math in high-dimensional space has deep parallels to algebraic systems.

. . .

3. The Algebraic View of Model Layers

Consider the diagram below, which reframes the LLM stack through algebra:

- **Helix encoding:** tokens spiral into structured sequences.
- **Algebraic lifts:** operations elevate tokens into higher algebraic dimensions.
- **Lexical layer:** raw tokens and subwords.
- **Embedding layer:** tokens mapped to continuous basis vectors.
- **Attention layers:** periodic structures aligned like rotations in algebraic fields.
- **Lexical beta at R:** an anchor point connecting language to algebraic reasoning.

This view makes explicit that **algebra isn't outside the model—it's inside the architecture itself.**

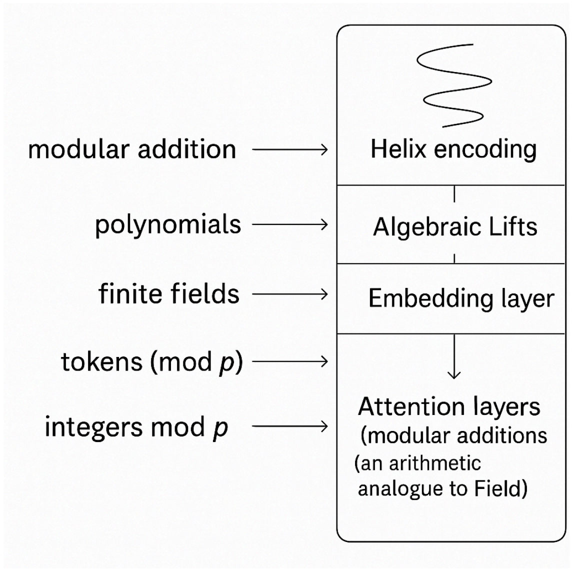

Figure 1—The Tokenization + Algebra pipeline.
Language tokens are lifted into algebraic space, passed through lexical, embedding, and attention layers, and finally grounded as lexical β at R. This shows how language models transform discrete symbols into structured algebraic representations.

. . .

4. Practical Implications for LLM Design

This algebraic perspective has design consequences:

- **Tokenizer optimization:** Subword tokenizers can act like "prime factor bases," reducing carry-over in sequence reasoning.
- **Algebra-aware architectures:** Embedding spaces constrained to algebraically inspired lattices could improve modular reasoning.
- **Error diagnosis:** Misalignments in arithmetic analogy often correlate with hallucination, suggesting new tools for interpretability.

By treating language as algebra, we can design models that reason more robustly and fail more predictably.

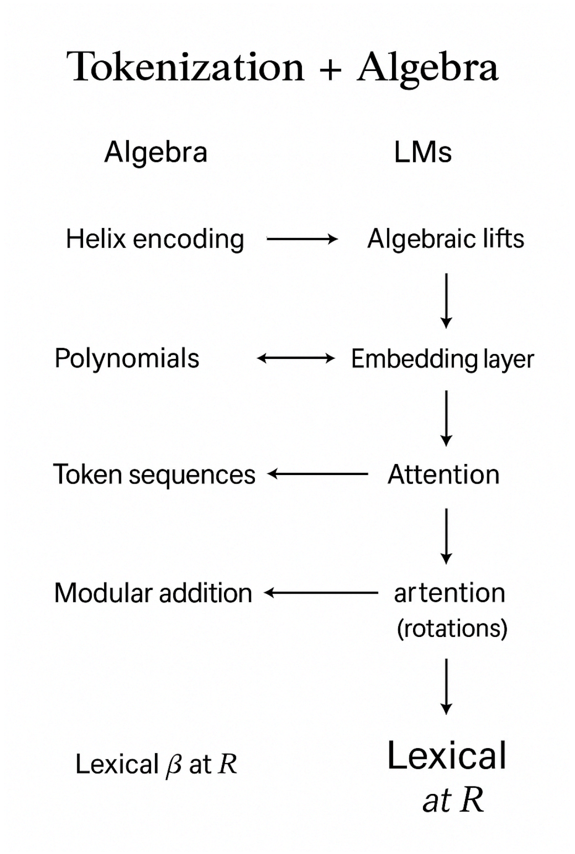

Figure 2—Algebraic parallels inside LLMs.
Token sequences map onto polynomials, attention mechanisms align with modular addition and rotations, and embeddings resemble finite fields. This diagram highlights how algebraic structures are embedded in the very architecture of large language models.

5. Toward a General Theory of Token Algebra

Across both research and practice, these ideas point toward a unifying principle:

LLMs behave as algebraic sequences.

What began as a clever engineering hack—tokenization—might actually be the doorway to a deeper theory of computation and cognition. By recognizing the algebraic structures that emerge within LLMs, we can better understand their power, their limits, and their path forward.

. . .

Closing

Language is numbers. Numbers are algebra. And algebra, hidden in plain sight, is what makes language models count, classify, and compute.

. . .

👉 *Follow me on Medium for more explorations at the intersection of algebra, language models, and cognition.*

. . .