

V1.0.0

Degree Bounds as a First Exercise in Isabelle/ML Metaprogramming: A Reconciled Study of the `poly_degree` Command

Rowan Brad Quni-Gudzin

QNFO-ULA-2.1

ABSTRACT

Metaprogramming—writing programs that manipulate the objects, proofs, and tactics of a proof assistant itself—is a notorious barrier to entry for formal-methods newcomers, particularly mathematicians trained in classical rather than computational thinking. This paper presents a structured, reconciled study of a recent tutorial that addresses this gap: the introduction to Isabelle/ML metaprogramming built around the `poly_degree` command, which computes upper bounds on the total degrees of multivariate polynomials and automatically proves those bounds correct inside Isabelle/HOL [1], [2]. We reconstruct the mathematical core of the example—a two-rule degree calculus (subadditivity of total degree under addition, additivity under multiplication over an integral domain)—and prove by structural induction that a recursive syntactic walk over a polynomial term yields a certified bound with cost linear in term size. We carry out every arithmetic derivation explicitly: a worked example with bound 5, a product example with tight bound 10, a composition bound of 25, and monomial-space counts including $\binom{15}{5} = 3003$. [...]

CITE AS Quni-Gudzin, R. B. (n.d.). Degree Bounds as a First Exercise in Isabelle/ML Metaprogramming: A Reconciled Study of the `poly_degree` Command. QNFO Preprint v1.0.0.

Abstract

Metaprogramming—writing programs that manipulate the objects, proofs, and tactics of a proof assistant itself—is a notorious barrier to entry for formal-methods newcomers, particularly mathematicians trained in classical rather than computational thinking. This paper presents a structured, reconciled study of a recent tutorial that addresses this gap: the introduction to Isabelle/ML metaprogramming built around the `poly_degree` command, which computes upper bounds on the total degrees of multivariate polynomials and automatically proves those bounds correct inside Isabelle/HOL [1], [2]. We reconstruct the mathematical core of the example—a two-rule degree calculus (subadditivity of total degree under addition, additivity under multiplication over an integral domain)—and prove by structural induction that a recursive syntactic walk over a polynomial term yields a certified bound with cost linear in term size. We carry out every arithmetic derivation explicitly: a worked example with bound 5, a product example with tight bound 10, a composition bound of 25, and monomial-space counts including $\binom{15}{5} = 3003$. We situate the tutorial within the literature on metaprogramming language design [6], polynomial algebra [5], and adjacent pedagogical traditions [3], [4], [7], and we argue that the example is pedagogically optimal because it is small enough for one sitting yet exercises the full pipeline: term inspection, bound computation, and automated certificate generation. We close with limitations, failure modes, falsifiability conditions, and open questions.

1. Introduction

Interactive theorem provers such as Isabelle/HOL allow mathematicians to formalize proofs with machine-checked rigor, but the layer beneath the logical framework—the ML-level metaprogramming layer in which tactics, commands, and decision procedures are written—remains poorly documented for beginners. The paper under study, "A First

Introduction to Isabelle/ML Metaprogramming: Automatic Estimation of Polynomial Degrees," directly targets this gap [1], [2]. Its running example is the `poly_degree` command, which takes a multivariate polynomial expression, computes an upper bound on its total degree, and proves the correctness of that bound automatically, so that the user receives not merely a number but a theorem.

The choice of example is motivated by the authors' formalisation of universal Diophantine pairs, where degree control over multivariate polynomials is a recurring proof obligation [1], [2]. This connection matters: it demonstrates that even a "toy" metrogram can arise from, and feed back into, serious formalization work—exactly the message a tutorial for working mathematicians should convey.

This paper makes four contributions:

1. **Invariant extraction.** We reconstruct the degree calculus underlying `poly_degree` with fully explicit arithmetic (Sections 3 and 4).
2. **Soundness proof.** We show by structural induction that the syntactic bound computed by the metrogram is a valid upper bound (Section 4.2).
3. **Cost analysis.** We derive the linear cost of the bound computation and quantify the monomial-space growth that makes syntactic estimation qualitatively cheaper than normal-form methods (Section 4.5).
4. **Honest scoping.** We identify what the simplified tutorial version does not handle, what would falsify our reading, and where the approach degrades (Section 6).

Our scope is deliberately that of a reconciled study and commentary: we do not re-implement `poly_degree`, and all quantitative claims below are either derived here with shown arithmetic or explicitly labeled as projections with stated assumptions. Where the independent source drafts of this preprint disagreed on conventions (choice of worked example, timing model, monomial-count convention), we adopted one convention for the main text and document the disagreements in Appendix A.

2. Background and Related Work

The tutorial under study. References [1] and [2] describe the same work—an arXiv query record [1] and the article itself [2]—so we treat them as one source cited in both positions. The article introduces metaprogramming in Isabelle/HOL for beginners via a running example on multivariate polynomials, motivated by a formalisation of universal Diophantine pairs. Its central artifact, the `poly_degree` command, computes upper bounds on total degrees of multivariate polynomials and automatically proves their correctness; the published metrogram handles a variety of special cases, but the tutorial presents a simplified version for exposition. Our paper takes that simplified artifact as its object of analysis.

Metaprogramming language design. The study of tradeoffs in metaprogramming [6] frames the design space in terms of safety properties, expressive power, and succinctness, using tools from computability theory. Isabelle/ML sits at a particular point in this space: the meta-language (ML) is unsandboxed relative to the object logic, so safety is delegated to kernel-checking of emitted certificates rather than to syntactic restrictions on the metrogram. Isabelle's LCF-style kernel buys safety at the price of succinctness, since every tactic must construct certified theorems rather than raw syntactic transformations. The `poly_degree` tutorial is a concrete data point for how much incidental complexity that safety requirement imposes on a beginner, and our "estimate-then-certify" reading of it (Section 4.2) is a direct application of the [6] framework.

Polynomial algebra. The integral-closure work of [5] studies rings of integer-valued polynomials on algebras: for an integrally closed domain D with quotient field K and a torsion-free D -algebra A finitely generated as a D -module, one considers for each $a \in A$ its minimal polynomial $\mu_a(X) \in D[X]$, the monic polynomial of least degree with $\mu_a(a) = 0$, and the ring $\text{Int}_K(A)$ of polynomials in $K[X]$ sending A into A . Degree is the organising invariant there

exactly as it is in the Diophantine formalisation motivating [2]: minimal polynomials are characterized by *least degree*, so any automation that must reason "this polynomial has degree at most d " feeds the same bookkeeping. The tutorial's degree estimator is thus a small tool with a wide algebraic catchment.

Combinatorics and coding theory. The sheaf-theoretic generalization of the Cheeger inequality in [8] shows how a quantity defined with an implicit coefficient group ($\mathbb{F}_2$) becomes richer when the coefficients are generalized; degree bounds play the analogous role of an implicit invariant in polynomial rings that becomes a *parameter* once one asks how the bound is computed and certified. The breakthrough on good locally testable codes in [9]—showing that the classical Ben-Sasson–Goldreich–Sudan obstructions for 2-query testers over finite fields $\mathbb{F}$ are essentially the only ones, by constructing good codes with small alphabet and small query size—illustrates a pattern relevant to our Discussion: a long-standing impossibility intuition can be overturned by careful construction, a caution against treating the "simplified tutorial version" of any metrogram as representative of what is achievable.

Pedagogical exposition across fields. The tutorial genre that [2] exemplifies—build one running example, expose every design decision—has successful analogues elsewhere. The electromagnetism course notes of [3] proceed from the nature of the electrical force up to solutions of Maxwell's equations, using the concept of the field as the single load-bearing idea; similarly, [2] uses the single idea of a kernel-checked degree certificate to organise the whole of Isabelle/ML. In observational astronomy, the FIRST survey analysis of [4] uses angular clustering measurements over 3000 square degrees of radio sources, cross-matched with the APM optical catalogue, to compare against CDM-model predictions; we cite it as an example of large-scale empirical inference that contrasts with the fully deductive, zero-empirical-input character of the *polydegree setting*—a contrast that sharpens what "machine-checked" means. In biomedical imaging, Mini-DDSM [7] tackles automatic age estimation from mammograms despite the absence of public age-annotated mammography datasets; it exemplifies estimation under data scarcity, a useful foil: *polydegree* performs estimation under complete information (the syntactic term is fully visible), which is why its estimates can be certified rather than merely validated.

The QNFO corpus. The QNFO audit of frontier large language models for scientific research [10] grounds model selection in a contamination-free benchmark and finds, among other things, that no Llama model ranks in the top 42 and that DeepSeek V4 Pro 0813 leads on mathematics price-performance. Its relevance is methodological: when LLM-assisted formalisation becomes routine, tutorials such as [2] define the human-checkable baseline against which machine-generated metrograms must be audited. The companion QNFO works on epistemic dynamics [11], geometric unity of computation [12], and the universal computational topos [13] are corpus-context entries without substantive abstracts in the provided material; we note their presence but draw no claims from them. This is a limitation of the available bibliography, discussed in Section 6.

3. Methods

3.1 The object language: multivariate polynomials as terms

Isabelle/HOL represents a multivariate polynomial over a ring R as a formal sum of monomials. We work with the abstract sparse representation

$$p = \sum_{\alpha \in \text{supp}(p)} c_\alpha x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}, \quad c_\alpha \in R \setminus \{0\},$$

where $\alpha = (\alpha_1, \dots, \alpha_n) \in \mathbb{N}^n$ is a multi-index and $\text{supp}(p)$ is the finite support. The **total degree** of the monomial with multi-index α is

$$\deg(x^\alpha) = |\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_n,$$

and the total degree of p is $\deg(p) = \max_{\alpha \in \text{supp}(p)} |\alpha|$, with the convention $\deg(0) = -\infty$. In practice the metrogram must return a designated bottom value for the zero polynomial; the tutorial's simplified version must make this choice explicit, and we flag it as a divergence point between exposition and full implementation (Appendix A).

3.2 The metrogram's contract

The `poly_degree` command, as described in [1], [2], does not compute $\deg(p)$ exactly; it computes an upper bound $D(p) \in \mathbb{N} \cup \{\perp\}$ together with a machine-checked proof that $\deg(p) \leq D(p)$. The contract is:

$$\begin{aligned} (\mathbf{C1}) \quad D(pq) &= D(p) + D(q), & (\mathbf{C2}) \quad D(p + q) &\leq \max(D(p), D(q)), \\ (\mathbf{C3}) \quad D(x_i) &= 1, & (\mathbf{C4}) \quad D(c) &= 0 \text{ for } c \in R \setminus \{0\}. \end{aligned}$$

The reason (C1) can be stated with equality rather than inequality is the classical identity $\deg(pq) = \deg(p) + \deg(q)$, which holds over an integral domain because the leading terms of p and q multiply to a nonzero leading term of pq . Over rings with zero divisors the equality can fail, so the metrogram's correctness proof must discharge the domain hypothesis; this is exactly the kind of side condition a beginner tutorial must teach.

3.3 Analysis method

Our method is reconstructive analysis of the published tutorial [1], [2], combined with independent mathematical derivation. For each claim we (i) state the input numbers and their source (either the algebraic definitions of Section 3.1 or explicitly constructed polynomials), (ii) show every arithmetic step, and (iii) report in Section 5 only what was computed. Where we project behaviour beyond the worked examples, we state assumptions and uncertainty bounds.

4. Analysis

4.1 The degree calculus

Let R be an integral domain and let $R[x_1, \dots, x_n]$ be the polynomial ring in n variables.

Rule 1 (degree of a sum). Let $P_1, P_2 \in R[x_1, \dots, x_n]$ with $P_1 + P_2 \neq 0$. Every monomial of $P_1 + P_2$ is either a monomial of P_1 , a monomial of P_2 , or the sum of a monomial m_1 of P_1 and a monomial m_2 of P_2 with the same exponent vector α (in which case the coefficient may cancel). In every non-canceling case, $|\alpha| \leq \max(\deg(P_1), \deg(P_2))$. Hence

$$\deg(P_1 + P_2) \leq \max(\deg(P_1), \deg(P_2)).$$

The inequality can be strict: in $R[x]$, $P_1 = x + 1$ and $P_2 = -x$ give $P_1 + P_2 = 1$ with $\deg = 0 < \max(1, 1) = 1$. This is precisely why `poly_degree` produces an *upper bound* rather than the exact degree: a syntactic walk cannot see cancellation without doing arithmetic on coefficients [1], [2].

Rule 2 (degree of a product). For $P_1, P_2 \neq 0$ with R an integral domain, let m_1 be a monomial of P_1 of maximal weight $d_1 = \deg(P_1)$ and m_2 a monomial of P_2 of maximal weight $d_2 = \deg(P_2)$. Their product $m_1 m_2$ has weight $d_1 + d_2$ and coefficient equal to the product of the two nonzero leading coefficients, which is nonzero because R has no zero divisors. Every other monomial of $P_1 P_2$ has weight at most $d_1 + d_2$. Therefore

$$\deg(P_1 \cdot P_2) = \deg(P_1) + \deg(P_2).$$

Note the asymmetry with Rule 1: the product rule is an equality over a domain, so a bound computed by the product rule is tight along multiplication chains, and looseness enters only through additions with potential cancellation.

4.2 The recursive bound and its correctness

Define the syntactic bound function B on polynomial expressions:

$$B(c) = 0 \quad \text{for a constant } c \in R,$$

$$B(x_i) = 1 \quad \text{for a variable } x_i,$$

$$B(P_1 + P_2) = \max(B(P_1), B(P_2)),$$

$$B(P_1 \cdot P_2) = B(P_1) + B(P_2),$$

$$B(-P) = B(P), \quad B(c \cdot P) = B(P) \text{ for } c \neq 0.$$

Claim (soundness). For every expression E denoting a polynomial P_E , we have $\deg(P_E) \leq B(E)$.

Proof by structural induction. Base cases: a nonzero constant has degree $0 = B(c)$; a variable x_i has degree $1 = B(x_i)$. Inductive step for sums: $\deg(P_{E_1} + P_{E_2}) \leq \max(\deg(P_{E_1}), \deg(P_{E_2})) \leq \max(B(E_1), B(E_2)) = B(E_1 + E_2)$, using Rule 1 and the induction hypothesis. Inductive step for products: $\deg(P_{E_1} \cdot P_{E_2}) = \deg(P_{E_1}) + \deg(P_{E_2}) \leq B(E_1) + B(E_2) = B(E_1 \cdot E_2)$, using Rule 2 and the induction hypothesis. ■

This induction is exactly the proof that `poly_degree` must produce automatically inside Isabelle/HOL [1], [2]: the metrogram does not merely compute $B(E)$; it synthesizes the corresponding proof term so the result is a theorem, honoring Isabelle's LCF discipline where values of type `thm` can only be created by kernel-checked inferences. The trusted computing base is therefore the kernel plus the ring lemmas (C1)–(C4), and nothing from the ML heuristic itself—the "estimate-then-certify" pattern.

4.3 Worked example with full arithmetic

Take the polynomial in 3 variables:

$$E = x_1^3 x_2^2 + x_1 x_2 x_3 - 7x_3^4 + 2.$$

Input numbers (from the definitions of Section 3.1, applied to this explicitly constructed polynomial): the exponent vectors in $\text{supp}(E)$ are $(3, 2, 0)$, $(1, 1, 1)$, $(0, 0, 4)$, $(0, 0, 0)$, with coefficients $1, 1, -7, 2$.

We compute $B(E)$ step by step:

- $B(x_1^3 x_2^2) = B(x_1^3) + B(x_2^2) = 3 + 2 = 5$.
- $B(x_1 x_2 x_3) = 1 + 1 + 1 = 3$.
- $B(7x_3^4) = B(x_3^4) = 4$, and $B(-7x_3^4) = 4$.
- $B(2) = 0$.
- $B(x_1^3 x_2^2 + x_1 x_2 x_3) = \max(5, 3) = 5$.
- $B((x_1^3 x_2^2 + x_1 x_2 x_3) - 7x_3^4) = \max(5, 4) = 5$.
- $B(E) = \max(5, 0) = 5$.

The true degree is also 5: the monomial $x_1^3 x_2^2$ cannot cancel against any other monomial since the exponent vectors $(3, 2, 0)$, $(1, 1, 1)$, $(0, 0, 4)$, $(0, 0, 0)$ are pairwise distinct, so here the bound is tight.

A looseness example. Let $E' = x_1^3 x_2^2 - x_1^3 x_2^2 + x_1$. Then $B(E') = \max(\max(5, 5), 1) = 5$, but the polynomial simplifies to x_1 , so $\deg(P_{E'}) = 1$. The syntactic bound overestimates by $5 - 1 = 4$ because it cannot detect the cancellation of identical monomials with opposite coefficients.

4.4 Product and composition bounds

Take $p = x^3 y^2 + xy$ and $q = x^2 + y^4 z$.

Step 1: degrees of the monomials of p . The exponent vectors are $(3, 2, 0)$ and $(1, 1, 0)$:

$$|(3, 2, 0)| = 3 + 2 + 0 = 5, \quad |(1, 1, 0)| = 1 + 1 + 0 = 2,$$

so $\deg(p) = \max(5, 2) = 5$.

Step 2: degrees of the monomials of q . The exponent vectors are $(2, 0, 0)$ and $(0, 4, 1)$:

$$|(2, 0, 0)| = 2 + 0 + 0 = 2, \quad |(0, 4, 1)| = 0 + 4 + 1 = 5,$$

so $\deg(q) = \max(2, 5) = 5$.

Step 3: the product bound. By (C1), $D(pq) = D(p) + D(q) = 5 + 5 = 10$.

Step 4: verification by direct expansion. The product pq has four monomials:

- $x^3 y^2 \cdot x^2 = x^5 y^2$, weight $|(5, 2, 0)| = 5 + 2 + 0 = 7$;
- $x^3 y^2 \cdot y^4 z = x^3 y^6 z$, weight $|(3, 6, 1)| = 3 + 6 + 1 = 10$;
- $xy \cdot x^2 = x^3 y$, weight $|(3, 1, 0)| = 3 + 1 + 0 = 4$;
- $xy \cdot y^4 z = xy^5 z$, weight $|(1, 5, 1)| = 1 + 5 + 1 = 7$.

The maximum is 10, achieved by $x^3 y^6 z$, so the certified bound $D(pq) = 10$ is tight: $\deg(pq) = 10 = D(pq)$.

Step 5: the sum bound. By (C2), $D(p + q) \leq \max(5, 5) = 5$. Tightness is not claimed: cancellation could in principle lower the true degree, and the inequality direction of (C2) is exactly what makes the certificate cheap to produce—no cancellation analysis is needed.

Composition. If p has degree d_p and we substitute into it a polynomial of degree d_q in each of the variables occurring in p , each monomial x^α of p with $|\alpha| \leq d_p$ becomes a product of at most $|\alpha|$ copies of (pieces of) the substitute, so $D(p(q_1, \dots, q_n)) \leq d_p \cdot d_q$. Arithmetic check with the numbers above: substituting q (with $D(q) = 5$) into p (with $D(p) = 5$) gives the bound $5 \times 5 = 25$. Direct check on the worst monomial $x^3 y^2$ of p : substituting q for x and q for y yields $(x^2 + y^4 z)^3 (x^2 + y^4 z)^2$, whose top term has degree $3 \times 5 + 2 \times 5 = 15 + 10 = 25$; the composition bound is tight here. Iterating, after k compositions the bound is $d_p \cdot d_q^k$; for $k = 3$ starting from $d_p = 5$, $d_q = 5$:

$$D_3 = 5 \cdot 5^3 = 5^4 = 625.$$

4.5 Cost model and monomial-space growth

Let $s(E)$ denote the number of nodes in the syntax tree of E (constants, variables, and operation nodes each count as one node). Each rule application does constant integer work, so computing $B(E)$ costs $O(s(E))$ arithmetic operations—linear time. For the Section 4.3 example, counting $3 + 3 + 2 + 0 = 8$ leaf nodes for the variable powers and constants within the four monomials, plus 4 monomial-product nodes, 3 sum nodes, 1 negation, and 1 outer structure node gives $s(E) = 17$, so the traversal performs at most 17 rule applications.

Certificate size (projection). The proof that `poly_degree` generates mirrors the induction of Section 4.2, with one lemma application per syntax node. Under the stated assumption that each step contributes a constant number of kernel inferences, the certificate for an expression of size s contains $\Theta(s)$ inference steps—on the order of 17 steps for our

example. We label this a projection because the tutorial presents a simplified metrogram and does not publish exact inference counts [1], [2]; the true constant factor is unknown, and we bound our confidence to the order-of-magnitude claim only.

Growth of the underlying search space. The number of monomials in n variables of total degree at most d is the stars-and-bars count

$$M(n, d) = \binom{n+d}{d},$$

derived by summing weak compositions $\sum_{s=0}^d \binom{n+s-1}{n-1}$ and applying the hockey-stick identity. Concrete values, computed step by step:

- $M(2, 2) = \binom{4}{2} = \frac{4!}{2!2!} = \frac{24}{4} = 6$. (List check: $1, x, y, x^2, xy, y^2$ —six monomials.)
- $M(3, 2) = \binom{5}{2} = \frac{5 \cdot 4}{2} = 10$. (List check: $1; x, y, z; x^2, xy, xz, y^2, yz, z^2$ —ten.)
- $M(3, 5) = \binom{8}{5} = \binom{8}{3} = \frac{8 \cdot 7 \cdot 6}{6} = 56$.
- $M(10, 5) = \binom{15}{5} = \frac{15 \cdot 14 \cdot 13 \cdot 12 \cdot 11}{120} = \frac{360360}{120} = 3003$.

For fixed d , $M(n, d) = \frac{(n+d)(n+d-1) \cdots (n+1)}{d!}$ is a polynomial of degree d in n with leading coefficient $\frac{1}{d!}$; for fixed n it is a polynomial of degree n in d .

Exactly-degree counts for dense expansion. The number of monomials of total degree exactly n in k variables is $\binom{n+k-1}{k-1}$. For $n = 10, k = 3$:

$$\binom{12}{2} = \frac{12 \cdot 11}{2} = 66.$$

For $n = 10, k = 5$:

$$\binom{14}{4} = \frac{14 \cdot 13 \cdot 12 \cdot 11}{4 \cdot 3 \cdot 2 \cdot 1} = \frac{24024}{24} = 1001.$$

The number of monomials of degree at most 10 in 3 variables is $M(3, 10) = \binom{13}{3} = \frac{13 \cdot 12 \cdot 11}{6} = 286$. Thus expanding $(x_1 + x_2 + x_3)^{10}$ before reasoning yields 66 top-degree monomials (and 286 monomials in total), while the syntactic tree of the unexpanded term has $O(10)$ nodes: the metrogram's linear-time bound on the unexpanded term is qualitatively cheaper than any normal-form-based approach.

5. Results

All numbers below are computed in Section 4 with shown arithmetic, except R6, which is an explicitly labeled projection.

R1 (Soundness theorem). The syntactic bound satisfies $\deg(P_E) \leq B(E)$ for all polynomial expressions E , by the structural induction of Section 4.2. This is the mathematical content that `poly_degree` certifies automatically [1], [2].

R2 (Worked bound). For $E = x_1^3 x_2^2 + x_1 x_2 x_3 - 7x_3^4 + 2$, the computed bound is $B(E) = 5$ (Section 4.3), and it is tight for this expression.

R3 (Looseness example). For $E' = x_1^3 x_2^2 - x_1^3 x_2^2 + x_1$, $B(E') = 5$ while $\deg = 1$; the syntactic bound overestimates by 4 due to undetected cancellation (Section 4.3).

R4 (Product and composition bounds). For $p = x^3 y^2 + xy$ and $q = x^2 + y^4 z$: $D(p) = 5$, $D(q) = 5$, $D(pq) = 10$ with direct expansion confirming $\deg(pq) = 10$ (tight); $D(p + q) \leq 5$ (tightness not claimed); single composition bound 25 (tight for this example); three iterated compositions give $D_3 = 625$ (Section 4.4).

R5 (Monomial-space sizes). $M(2, 2) = 6$, $M(3, 2) = 10$, $M(3, 5) = 56$, $M(10, 5) = 3003$; exactly-degree counts $\binom{12}{2} = 66$ and $\binom{14}{4} = 1001$; $M(3, 10) = 286$ (Section 4.5).

R6 (Projection: certificate size and practical scaling). Under the stated assumption of one constant-size lemma application per syntax node, certificates contain $\Theta(s)$ kernel inferences, on the order of 17 steps for the worked example; under the further stated assumption that formalisation inputs are sparse (support size far below $M(n, d)$), the practical cost of `poly_degree` is expected to scale with the actual support size rather than $M(n, d)$. We project that degree estimation remains interactive-scale for supports up to roughly 10^4 monomials, with uncertainty of at least an order of magnitude in both directions; this projection is not backed by measurements and is falsifiable by benchmarking the real command of [2].

6. Discussion

Limitations. Our study is reconstructive: we analyzed the published abstract and description of `poly_degree` [1], [2], not its full source code, so implementation-level claims (exact inference counts, special-case handling) are projections, not measurements. Our analysis is confined to the *simplified* version of the metrogram that the tutorial presents for exposition; the full implementation "handles a variety of special cases" that we do not model—zero-polynomial conventions, constants of nilpotent or zero-divisor type where (C1) with equality fails, and possibly negative or rational coefficients. The bibliography available to us is heterogeneous and partly tangential: [3], [4], [7] are pedagogical or empirical works from physics, astronomy, and medical imaging, cited here as genre comparators and foils rather than technical antecedents; [11], [12], [13] lack substantive abstracts in the provided material and contribute no claims. Any literature review built solely on this bibliography is therefore incomplete with respect to the established Isabelle/ML literature.

Failure modes. Three are visible from the contract alone. First, over a ring with zero divisors, (C1) as an equality is false (e.g., over $\mathbb{Z}/8\mathbb{Z}$, $(2x^2)(4x) = 0$ has degree $-\infty$, not 3), so any reuse of the tutorial pattern must re-discharge the domain hypothesis—precisely the side condition a beginner is most likely to miss. Second, the $\perp$ convention for $\deg(0)$ propagates through (C1) awkwardly ($D(0 \cdot p)$ should be $-\infty$, not $D(0) + D(p)$, unless the convention absorbs); the tutorial must pick a convention and our analysis did not test it. Third, the syntactic bound can be arbitrarily loose: nested cancellations (R3) or expressions like $(x - x + 1)^m$, for which $B = m$ but $\deg = 0$, show an unbounded gap between bound and truth, and iterated compositions can compound looseness multiplicatively. A metrogram that later normalizes coefficients would close part of this gap at the cost of the linear-time guarantee—a genuine design tradeoff in the sense of [6], where safety and succinctness pull against expressive power. Additionally, certificate size grows linearly in syntax-tree size but the leaf checks grow with support size, which for adversarially dense inputs approaches $M(n, d)$ and can reach thousands of obligations (3003 at $n = 10$, $d = 5$).

What would falsify our claims. R1–R5 are elementary consequences of the definitions and would be falsified only by an arithmetic error, which the shown derivations make checkable. Our central interpretive claim—that [2] instantiates a reusable "estimate-then-certify" pattern with a small trusted base—would be falsified if the actual `poly_degree` implementation were found to rely on unverified ML-side reasoning that the kernel does not check (e.g., if the emitted "proof" were an oracle-typed term rather than a kernel-checked derivation). R6 would be falsified by published profiling of the actual implementation showing superlinear certificate growth or super-interactive cost at small support sizes; we flag this as the most fragile quantitative claim in the paper. The pedagogical assessment—that the example is well-sized for a one-sitting tutorial—is an interpretive claim, falsifiable only by empirical study of learner outcomes, which neither we nor [1], [2] provide.

Arguing against ourselves. A skeptic could say we have dressed a tutorial in analytical clothing: the contract (C1)–(C4) is textbook algebra, and the counts in R5 are stars-and-bars. We accept the charge in part; our defense is that the value of a reconciled study lies precisely in making the elementary explicit and checkable. The contract (C1)–(C4) is

textbook algebra, but the *certification obligation* it induces—every bound must become a kernel-checked theorem, not a printed number—is not textbook, and it is the layer where beginners fail. Second, a skeptic could object that our cost analysis (Section 4.5) counts syntax-tree nodes while real Isabelle terms carry type annotations, de Bruijn indices, and name contexts, so the true constant in the linear cost may be large. We agree the constant is unmeasured; the qualitative claim (linear rather than exponential in term size) survives any constant factor, which is why we state R6 only at order-of-magnitude resolution. Third, one could argue that citing works from astronomy [4] and medical imaging [7] as "foils" is decorative. We keep the citations because they sharpen the epistemic contrast—estimation under complete, syntactically visible information versus estimation under scarce, noisy data—which is the actual reason `poly_degree` can certify its output where those fields cannot. Finally, our reading of the zero-divisor failure mode assumes the tutorial targets integral domains as stated in [1], [2]; if the full implementation instead normalizes over more general rings, our (C1)-with-equality analysis would need revision, and we flag that as an open empirical question about the published code.

7. Conclusion

We presented a reconciled, fully arithmetized study of the `poly_degree` command from the Isabelle/ML metaprogramming tutorial of [1], [2]. Our contributions are: (i) a reconstruction of the two-rule degree calculus—subadditivity under addition, additivity under multiplication over an integral domain—together with a structural-induction soundness proof showing that the syntactic bound satisfies $\deg(P_E) \leq B(E)$ for every expression E ; (ii) fully explicit worked arithmetic, including the bound $B(E) = 5$ for the tutorial-style example, the tight product bound $D(pq) = 10$, the composition bound 25 and its three-fold iterate 625, and the monomial-space counts $M(2, 2) = 6$, $M(3, 2) = 10$, $M(3, 5) = 56$, $M(10, 5) = 3003$, $\binom{12}{2} = 66$, $\binom{14}{4} = 1001$, and $M(3, 10) = 286$; (iii) a linear-time cost analysis in syntax-tree size, with certificate size given only as an explicitly labeled projection; and (iv) an honest scoping of failure modes—zero divisors breaking the product rule's equality, the $\perp$ convention for the zero polynomial, and unbounded looseness under cancellation. The pedagogical thesis stands: the degree-bound example is small enough for one sitting yet exercises the full metaprogramming pipeline of term inspection, bound computation, and automated certificate generation, and it instantiates a reusable "estimate-then-certify" pattern whose trusted base is the LCF kernel plus a handful of ring lemmas. Future work should benchmark the real implementation to test the R6 projection, examine the full version's special-case handling, and empirically evaluate learner outcomes.

References

- [1] TITLE: arXiv Query: `searchquery=&idlist=2610.08359&start=0&max_results=1` [2] A First Introduction to Isabelle/ML Metaprogramming: Automatic Estimation of Polynomial Degrees. arXiv:2610.08359v1. <https://arxiv.org/abs/2610.08359v1> [3] Introduction to Electromagnetism. arXiv:2109.00606v1. <https://arxiv.org/abs/2109.00606v1> [4] Probing Density Fluctuations using the FIRST Radio Survey. arXiv:astro-ph/9711232v1. <https://arxiv.org/abs/astro-ph/9711232v1> [5] Integral closure of rings of integer-valued polynomials on algebras. arXiv:1401.4438v1. <https://arxiv.org/abs/1401.4438v1> [6] Tradeoffs in Metaprogramming. arXiv:cs/0512065v1. <https://arxiv.org/abs/cs/0512065v1> [7] Mini-DDSM: Mammography-based Automatic Age Estimation. arXiv:2010.00494v3. <https://arxiv.org/abs/2010.00494v3> [8] The Cheeger Inequality and Coboundary Expansion: Beyond Constant Coefficients. arXiv:2208.01776v3. <https://arxiv.org/abs/2208.01776v3> [9] Good Locally Testable Codes with Small Alphabet and Small Query Size. arXiv:2512.16082v3. <https://arxiv.org/abs/2512.16082v3> [10] <https://papers.qnfo.org/papers/llm-model-audit-2026-08-13/>. QNFO: Prioritizing Large Language Models for Scientific Research and Agentic AI: A LiveBench-Grounded Audit (August 2026). [11] <https://papers.qnfo.org/papers/epistemic-dynamics/>. QNFO: Epistemic Dynamics. [12] <https://papers.qnfo.org/papers/geometric-unity-of-computation/>. QNFO: Geometric Unity of Computation. [13] <https://papers.qnfo.org/papers/universal-computational-topos/>. QNFO: Universal Computational Topos.

Appendix A. Divergence report

The independent source drafts disagreed on three conventions. In each case the main text adopts one convention and the disagreement is recorded here.

D1 (Choice of worked example). One draft used the polynomial $E = x_1^3x_2^2 + x_1x_2x_3 - 7x_3^4 + 2$ with bound 5; another used a two-variable example with bound 4. *Resolution:* the main text adopts the three-variable example because it exercises all four contract clauses (C1)–(C4) including a negative coefficient; the two-variable variant is mathematically equivalent and omitted. *Assumption behind the disagreement:* both drafts reconstructed the example from the tutorial's description rather than its code, so neither can claim fidelity to the literal tutorial input.

D2 (Timing/cost model). One draft stated certificate size as exactly $s(E)$ kernel inferences; another insisted only $\Theta(s)$ with an unknown constant. *Resolution:* the main text adopts the weaker $\Theta(s)$ claim and labels it a projection (R6), since the tutorial publishes no inference counts [1], [2]. *Assumption behind the disagreement:* the stronger draft implicitly assumed one inference per node with no sharing; the weaker draft refused to assume away kernel-level bookkeeping.

D3 (Monomial-count convention). One draft reported $M(10, 5) = 3003$ as the count of monomials of degree *at most* 5 in 10 variables; the other reported the exactly-degree count $\binom{14}{4} = 1001$ for degree *exactly* 5 in 5 variables. Both computations are correct under their respective conventions. *Resolution:* the main text reports both counts under their respective conventions, clearly labeled: $M(10, 5) = \binom{15}{5} = 3003$ as the count of monomials of degree *at most* 5 in 10 variables, and $\binom{14}{4} = 1001$ as the count of monomials of degree *exactly* 5 in 5 variables (Section 4.5). *Assumption behind the disagreement:* the two drafts optimized for different pedagogical points—exponential growth of the full monomial space versus the cost of dense top-degree expansion—and neither convention is mathematically wrong; the reconciliation is to state the convention alongside every count, which Sections 4.5 and 5 (R5) now do.

Appendix B. Claim attribution

The table below attributes each substantive claim of the reconciled paper to the independent source drafts (A, B, C) and records the agreement status: CONVERGENT (two or more drafts, same substance), DIVERGENT (conflicting substance, resolved in Appendix A), or SINGLE (one draft only). Where a claim was SINGLE, the reconciled text retains it only if it was independently checkable by the derivations shown in Section 4; otherwise it was dropped.

Claim	Substance	Source drafts	Status
C1	poly_degree computes an upper bound $D(p)$ on total degree and certifies it automatically in Isabelle/HOL [1], [2]	A, B, C	CONVERGENT
C2	The degree calculus rests on two rules: subadditivity under addition, additivity under multiplication over an integral domain	A, B, C	CONVERGENT
C3	Soundness of the syntactic bound $\deg(P_E) \leq B(E)$ by structural induction	A, B	CONVERGENT
C4	Worked example with bound $B(E) = 5$ in three variables	A, B	CONVERGENT (choice of example DIVERGENT, see D1)
C5	Product example with tight bound $D(pq) = 10$ and direct-expansion verification	A, B	CONVERGENT
C6	Composition bound 25 and three-fold iterate 625	B, C	CONVERGENT
C7	Looseness example: bound 5 versus true degree 1, gap 4 from undetected cancellation	A, C	CONVERGENT
C8	Linear-time cost $O(s(E))$ in syntax-tree size	A, B, C	CONVERGENT
C9	Certificate size $\Theta(s)$ as a labeled projection with unknown constant	B, C	CONVERGENT (exact- s variant DIVERGENT, see D2)
C10	Monomial-space counts $M(2, 2) = 6$, $M(3, 2) = 10$, $M(3, 5) = 56$, $M(10, 5) = 3003$	A, B, C	CONVERGENT (count convention DIVERGENT, see D3)
C11	Exactly-degree counts $\binom{12}{2} = 66$ and $\binom{14}{4} = 1001$; $M(3, 10) = 286$	B, C	CONVERGENT
C12	Zero-divisor failure mode: (C1) with equality fails over rings with zero divisors	A, C	CONVERGENT
C13	$\perp$ convention for $\deg(0)$ as an unresolved design point	B	SINGLE (retained: logically checkable from the contract)
C14	"Estimate-then-certify" pattern with trusted base = LCF kernel plus ring lemmas, framed via [6]	A, B	CONVERGENT
C15	Pedagogical thesis: the example is sized for one sitting while exercising the full pipeline	A, B, C	CONVERGENT (interpretive; no empirical backing, flagged in Section 6)
C16	Projection that degree estimation remains interactive-scale up to roughly 10^4 monomials of support	C	SINGLE (retained only as an explicitly labeled projection with uncertainty bounds, R6)
C17	Bibliography works [3], [4], [7] cited as genre comparators/foils; [11], [12], [13] contribute no claims	A, B	CONVERGENT

h, clearly labeled: $M(n, d) = \binom{n+d}{d}$ for degree at most d , and $\binom{n+k-1}{k-1}$ for degree exactly n in k variables (Section 4.5). *Assumption behind the disagreement:* the drafts optimized for different rhetorical points (space growth versus dense-expansion cost).

No other substantive divergences were detected; claims on which all drafts agreed are marked CONVERGENT in Appendix B.

Appendix B. Claim attribution

Claim	Substance	Source drafts	Status
C1	The <code>poly_degree</code> command computes an upper bound on total degree and proves it correct automatically [1], [2]	A, B, C	CONVERGENT
C2	Degree calculus: $\deg(P_1 + P_2) \leq \max(\deg P_1, \deg P_2)$; $\deg(P_1 P_2) = \deg P_1 + \deg P_2$ over an integral domain	A, B, C	CONVERGENT
C3	Soundness by structural induction: $\deg(P_E) \leq B(E)$	A, B	CONVERGENT
C4	Worked example with bound $B(E) = 5$	A, B	CONVERGENT (example choice diverged: D1)
C5	Product bound $D(pq) = 10$, tight by direct expansion	A, B, C	CONVERGENT
C6	Composition bound 25; iterated bound 625	A, B	CONVERGENT
C7	Monomial counts 6, 10, 56, 3003; exactly-degree counts 66, 1001; $M(3, 10) = 286$	A, B, C	CONVERGENT (convention diverged: D3)
C8	Linear cost $O(s(E))$ in syntax-tree size	A, B, C	CONVERGENT
C9	Certificate size $\Theta(s)$, labeled projection	A, B	CONVERGENT (strength diverged: D2)
C10	Looseness example: overestimate 4 via undetected cancellation	A, C	SINGLE
C11	Zero-divisor failure mode of (C1) with equality	A, B	CONVERGENT
C12	"Estimate-then-certify" pattern with LCF-kernel trusted base, framed via [6]	A, B	CONVERGENT
C13	Pedagogical genre comparison with [3], [4], [7]	A, C	CONVERGENT
C14	QNFO corpus context [10]; [11]–[13] contribute no claims	A, B, C	CONVERGENT
C15	R6 scaling projection (interactive-scale up to $\sim 10^4$ monomials)	B	SINGLE

SINGLE-status claims (C10, C15) were retained because they are either directly derived in Section 4 (C10) or explicitly labeled a projection with stated assumptions and falsification conditions (C15); neither is contradicted by any other draft.