The polynomial definition: functions with a representative, not function types

August 24, 2026

1 The paper's definition

In [JNV+20], an m-variate polynomial over $\mathbb{F}_q$ is defined as a *function* $g : \mathbb{F}_q^m \to \mathbb{F}_q$ of the form

$$g(x_1, \ldots, x_m) = \sum_{i_1,\ldots,i_m} c_{i_1,\ldots,i_m} \cdot x_1^{i_1} \cdots x_m^{i_m},$$

with coefficients $c_{i_1,\ldots,i_m} \in \mathbb{F}_q$.[1] Individual degree d means $i_1, \ldots, i_m \leq d$ for every nonzero coefficient. The set

$$\mathcal{P}(m, q, d) \subset \{\, \mathbb{F}_q^m \to \mathbb{F}_q\}$$

is then the set of functions that admit such a representation.[2] This is the standard convention in theoretical computer science: "polynomial" means a function that can be written as an algebraic polynomial.

2 The formal definition

The formalization defines a low-individual-degree predicate[3] on a function $g : \mathbb{F}_q^m \to \mathbb{F}_q$ as

$$\mathrm{LowDeg}_d(g) := \exists\, p \in \mathbb{F}_q[x_1, \ldots, x_m], (\forall i,\ \deg_{x_i}(p) \leq d) \wedge g = \mathrm{eval}(p).$$

Here p ranges over the algebraic polynomial ring over the chosen finite field,[4] and eval is its coefficient-wise evaluation map to a function $\mathbb{F}_q^m \to \mathbb{F}_q$. The predicate therefore asserts exactly the paper's condition: g admits an algebraic polynomial representative of the right degree.

[1] See `references/ldt-paper/introduction.tex`, line 3.

[2] See `references/ldt-paper/preliminaries.tex`, lines 92–93.

[3] The declaration is named HasLowIndividualDegree and appears in `MIPStarRE/LDT/Basic/LinePolynomials.lean`, line 20.

[4] This is the abbreviation PolynomialModel, which expands to MvPolynomial (Fin m) (Scalar params) in `MIPStarRE/LDT/Basic/ParametersBase.lean`, line 300.

The earlier blueprint used a function-type representation for $\mathcal{P}(m, q, d)$, treating low-degree polynomials as a subtype of the function space. That representation has been corrected to the $\exists$ predicate above, which matches the paper's definition exactly.[5]

3 Why algebraic polynomials?

The paper's proofs (and the formalization) require algebraic polynomials for the Schwartz–Zippel lemma.[6] The Mathlib statement operates on algebraic polynomials, not on arbitrary functions. The low-individual-degree predicate is the natural bridge: it takes a function g and extracts the algebraic polynomial p that witnesses its low individual degree, which enables the application of Schwartz–Zippel.

The measurement families G, A, B, L in the formalization carry outcomes indexed by *polynomials*, and the evaluation maps $g \mapsto G_g$ respect the algebraic polynomial structure. This design closely follows the paper, where measurement outcomes are indexed by $g \in \mathcal{P}(m, q, d)$ and the Schwartz–Zippel step uses algebraic properties of the polynomials.

4 Conclusion

There is no divergence. The paper defines polynomials as functions with a polynomial representative; the formalization defines the low-individual-degree predicate as the existence of an algebraic polynomial representative. The two are mathematically equivalent. The earlier blueprint, which used function types directly, has been corrected to the $\exists$ predicate, eliminating the only discrepancy.

References

[JNV+20] Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. Quantum soundness of the classical low individual degree test, 2020. arXiv:2009.12982.

[5] The blueprint correction is visible in `blueprint/src/chapter/ch02_test.tex` and `blueprint/src/chapter/ch03_preliminaries.tex`, where $\mathcal{P}(m, q, d)$ is now the set of functions representable as an algebraic polynomial of individual degree $\leq d$.

[6] Labelled lem:schwartz-zippel-total-degree in the paper, and available in Mathlib as MvPolynomial.schwartzZippel.