A p-adic Perspective on Low-Bit Training of Neural Networks

author names withheld

Under Review for the Workshop on High-dimensional Learning Dynamics, 2026

Abstract

We investigate low-bit neural network training from a p-adic perspective. In extreme low-bit regimes, the set of representable values is so small that gradient descent operates on an essentially discrete domain, making continuous analysis inadequate. This observation motivates us to model a neural network as a polynomial system over the integers modulo p^N. Activations and losses are replaced by piecewise polynomial approximations, and training is recast as finding roots of the resulting system. Hensel's lemma provides an iterative procedure for lifting seed roots digit-by-digit to the required precision. We formalize this approach and demonstrate its feasibility on linear regression and shallow polynomial networks.

1. Introduction

Large neural models are now routinely compressed, adapted, and even trained in numerical formats whose bit-width is far below that assumed by standard floating-point optimization theory [1–8]. The usual analysis of low-bit training often treats quantization as truncation, rounding, or an additive perturbation whose distribution is approximated by a continuous noise model. This point of view is useful at moderate precision, but it becomes less informative in extreme low-bit regimes. A 4-bit representation has at most $2^4 = 16$ codes before accounting for scale choices. The training state is therefore not a small continuous perturbation of a real-valued state: it belongs to a finite set whose arithmetic and representability constraints must be taken seriously.

This paper studies one algebraic way to make that discreteness explicit. After fixing a finite representation, model values are rational numbers. If the loss, activations, and model operations are polynomial, or are replaced by explicitly specified piecewise polynomial approximations, then the stationarity equations can be cleared of denominators and written as an integer polynomial system. Reducing this system modulo p^N gives a finite algebraic candidate problem.

We use p-adic methods in this formulation only through modular arithmetic and Hensel lifting for polynomial equations over finite residue rings. A root modulo p can sometimes be lifted digit by digit to a compatible root modulo p^N, provided that the relevant derivative or Jacobian is nonsingular modulo p. An exact finite-bit stationary point of the integerized system gives a modular root. The converse is false in general: a modular root is only a candidate residue class. It must be lifted, reconstructed as an integer or rational representative when appropriate, and verified against the original finite formulation.

The contributions are scoped to candidate generation and verification: we reduce finite-bit piecewise-polynomial training objectives to integer stationarity systems modulo p^N; make explicit that modular roots are candidates, not certified optima; verify lift-reconstruct-check on synthetic linear regression; and show on a small polynomial MLP that singular Jacobians, not root existence alone, are a practical bottleneck.

2. Method

We start from the standard empirical risk minimization objective

$$\min_{\boldsymbol{w}} \ \mathcal{L}(\boldsymbol{w}; X, Y) = \min_{\boldsymbol{w}} \ \sum_{k} \ell(\boldsymbol{w}; x_k, y_k), \tag{1}$$

where (x_k, y_k) are input-target pairs and X, Y collect the dataset or batch. Our goal is not to run gradient descent in $\mathbb{Q}_p$. Instead, we replace a finite-bit polynomial surrogate of (1) by an integer stationarity system and study its modular roots.

Modular reduction. The direct substitution of real parameters by p-adic parameters is not the claim of this paper. Gradients and local descent are tied to the topology of the space, and the p-adic topology is not a small perturbation of the Euclidean one. We therefore use p-adic and modular tools only after an algebraic integerization step.

Hensel's lemma (reviewed in Appendix A) is useful for polynomial equations because it can lift a seed root modulo p to a compatible root modulo p^N when the appropriate derivative or Jacobian is invertible modulo p. Under the two assumptions below, this gives a candidate-generation procedure for the polynomial surrogate of (1), followed by rational reconstruction and verification in the original finite formulation.

Assumption 2.1. *(Piecewise polynomial approximation)* The model, activation functions, and loss are either polynomial on the representable domain or are replaced by explicitly specified piecewise polynomial approximations with rational coefficients.

Assumption 2.2. *(Finite bitwidth)* Weights, inputs, labels, and coefficients used in the finite formulation [9] are represented with finite bitwidth. We model them by integer code variables with fixed rational scales; after substituting the scales, the coefficients are rational.

For the polynomial surrogate, stationary candidates satisfy $F(\boldsymbol{w}) = \partial \frac{\mathcal{L}}{\partial} \boldsymbol{w} = 0$. This is a necessary condition for a smooth local minimizer, but it is not sufficient: stationary points may be saddles or maxima, and finite-domain local optimality must be verified separately. Under Assumption 2.1, F is a vector-valued polynomial system.

After fixing the polynomial pieces and finite-bit representation, the coefficients of the surrogate stationarity system are rational. Let D clear all coefficient and scale denominators and define $\tilde{F}(\boldsymbol{w}) = DF(\boldsymbol{w}) \in \mathbb{Z}[\boldsymbol{w}]$, where $\boldsymbol{w}$ denotes integer codes. For a chosen prime p and precision N, we study the modular candidate system

$$\tilde{F}(\boldsymbol{w}) \equiv 0 \pmod{p^N}. \tag{2}$$

Thus, exact finite-bit stationary points of (1) reduce to solutions of (2) after clearing denominators. The converse is not true in general since a residue satisfying (2) is only a candidate and must be lifted, rationally reconstructed when appropriate, and verified in the original finite formulation. This one-way reduction is stated in Proposition 2.3 (see a proof sketch in Appendix B).

Proposition 2.3 *(One-way modular reduction)* With the assumptions 2.1 and 2.2, any bounded finite-bit code vector that satisfies the integer stationarity equations for the encoded surrogate of (1) gives a residue class satisfying (2). A solution of (2) need not be a stationary point or a local minimizer of the original finite problem.

Algorithm. Solving (2) directly is hard. We split the task into seed-root search modulo p and Hensel lifting to the target precision p^N (see Listing 1). In the lifting step, write F for the integerized system $\tilde{F}$ and let $\boldsymbol{w}_n \in \mathbb{Z}/p^n\mathbb{Z}$ satisfy $F(\boldsymbol{w}_n) \equiv 0 \pmod{p^n}$. Taylor expansion modulo p^{n+1} gives

$$F(\boldsymbol{w}_n + p^n \delta_n) \equiv F(\boldsymbol{w}_n) + p^n G(\boldsymbol{w}_n)\delta_n \pmod{p^{n+1}}, \tag{3}$$

where $G = \partial F / \partial \boldsymbol{w}$ is the Jacobian. The next correction $\delta_n \in (\mathbb{Z}/p\mathbb{Z})^d$ must solve

```
def candidates(loss, p, digits):
  F = integer_gradient(loss)
  J = jacobian(F)
  for r in initial_roots(F, p):
    if rank(J(r) % p) < len(F):
      continue
    rN = hensel(F, J, r, p, digits)
    if rN is None:
      continue
    w = reconstruct(rN)
    if verify_original_problem(w):
      yield w
```

```
def hensel(F, J, r, p, digits):
  for n in range(1, digits):
    A = J(r) % p
    b = (-F(r) // p**n) % p
    if det(A) % p == 0:
      return None
    delta = solve_mod_p(A, b, p)
    r = r + p**n * delta
  return r
```

Listing 1: Pseudocode of the modular candidate pipeline. The pseudocode covers the nonsingular square-system case. Singular seeds are treated as failures, although they may admit non-unique lifts.

$$G(\boldsymbol{w}_n)\delta_n \equiv -\frac{F(\boldsymbol{w}_n)}{p^n} \pmod{p}, \tag{4}$$

where $F(\boldsymbol{w}_n)$ is divisible by p^n by the induction hypothesis.

Thus each iteration solves a linear system over $\mathbb{F}_p$. In the nonsingular square-system case this system has a unique solution. If the Jacobian is singular, ordinary Hensel lifting does not certify a unique lift; this is a central limitation discussed in Section 3.

3. Limitations

The formulation in this paper is intentionally limited. It gives an algebraic candidate-generation procedure, not a complete replacement for optimization or a proof that modular roots are optima of the original finite objective. Further discussion of seed search and lifting conditions is in Appendix C.1 and Appendix C.2.

Certification. Solving the modular system modulo p^N gives candidate residue classes. Rational reconstruction then attempts to select low-height integer or rational representatives. A candidate becomes evidence for the original finite formulation only after it satisfies the original equations, quantization constraints, rounding assumptions, and finite-domain verification criterion. A modular root of a gradient system is therefore not by itself a certified optimum of the finite objective.

Seed Search. Hensel lifting starts from a root modulo p, but finding such roots is a hard finite-field search problem for neural networks. Exhaustive search scales as p^d in the number of parameters, and continuous training followed by quantization does not automatically produce an exact modular root. Initial-root discovery should therefore be treated as a central research problem rather than an implementation detail.

Lifting Conditions. Ordinary square-system Hensel lifting requires the Jacobian of the integer polynomial system to have full rank modulo p at the seed. If this condition fails, a root modulo p may have no lift, multiple lifts, or a positive-dimensional family of lifts. The small polynomial MLP experiment in Section 4.2 illustrates this problem: many raw modular roots were found, but all observed roots were singular and therefore unusable as ordinary Hensel seeds.

4. Experiments

The experiments are designed to test the algebraic bridge rather than to benchmark training performance. We ask whether a modular root can be found, whether the nonsingular Hensel condition

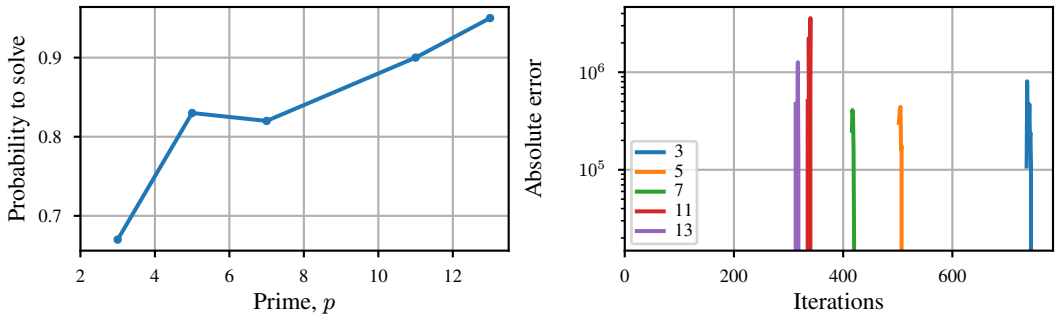

Figure 1: Synthetic linear-regression sweep. Left: fraction of 100 random seeds for which the modular system had a nonsingular initial root and rational reconstruction verified the exact normal equations. Right: mean absolute stationarity residual after rational reconstruction during Hensel lifting. The experiment uses fixed-point scale 2^4, $d = 32$, $n = d^2$ samples, and 1000 Hensel iterations.

holds, whether the lifted residue can be reconstructed, and whether the reconstructed point satisfies the original integer stationarity equations.

All reported runs use exact integer or finite-field arithmetic on CPU. The linear-regression sweep reports 100 independent random draws for each prime. The MLP rows report the number of residues visited explicitly; capped rows should be read as search traces, not as exhaustive absence claims.

4.1. Linear Regression

Linear regression gives a controlled setting where every step of the pipeline can be verified exactly. With inputs X and targets Y, the objective

$$\mathcal{L}(A) = \frac{1}{2}\|Y - AX\|_F^2. \tag{5}$$

has stationarity equations

$$AXX^\top = YX^\top. \tag{6}$$

After vectorization, this is a square linear system

$$(I_m \otimes XX^\top)\,\mathrm{vec}(A) = \mathrm{vec}(YX^\top), \tag{7}$$

whose Jacobian is constant. For a prime p, the nonsingular Hensel condition therefore reduces to invertibility of $XX^\top$ modulo p. A successful run finds the solution modulo p, lifts it to p^N, reconstructs a rational representative, and verifies the normal equations over $\mathbb{Q}$.

We generated fixed-point synthetic data from rounded Gaussian features and a rounded teacher model with scale 2^4. For each prime, we ran 100 random seeds with $d = 32$, $n = d^2$ samples, and 1000 Hensel iterations. A run is counted as solved only when rational reconstruction gives a point satisfying the integer normal equations exactly (see Table 3 in Appendix D).

The result is intentionally modest. Linear regression does not test nonlinear neural-network training, but it does test the full algebraic pipeline in a case where success and failure are unambiguous. The failure at $p = 2$ in this sweep is also informative: no trial had a nonsingular initial system modulo 2, so ordinary Hensel lifting could not certify a unique lift. For odd primes, the observed failures are exactly the cases where the nonsingularity condition does not hold.

We also used the `scikit-learn` Diabetes dataset [10, 11] as a real-data sanity check for the same fixed-point normal-equation solver. In local runs, the lifted residues were reconstructed and verified after roughly 350 Hensel steps. We do not use this as the main empirical evidence because the synthetic sweep above gives repeated trials with controlled dimensions and verification statistics.

Surrogate	p	Seeds	Search	Visited	Raw Roots	Nonsing.
quad.	2	0	$\mathbb{F}_2^{12}$	full	960	0
quad.	3	0–10	$\mathbb{F}_3^{12}$	full each	408–5307	0
quad.	5	42/223/228	$\mathbb{F}_5^{12}$	full each	20235–27510	0
quartic	5	1	$\mathbb{F}_5^9$ cap	$10k/5^9$	435	0
quartic	7	1	$\mathbb{F}_7^9$ cap	$1k/7^9$	41	0

Table 1: Seed-search traces for a small polynomial MLP. "Raw roots" satisfy $F(w) \equiv 0 \pmod{p}$ before the nonsingularity filter. Every observed raw root had singular Jacobian and was rejected as an ordinary Hensel seed. "Full" means all residues in the displayed search space were visited; capped $\mathbb{F}_p^9$ rows enumerate first-layer weights and solve last-layer weights when possible.

4.2. Polynomial MLP Seed Search

To test whether the same nonsingular-seed requirement survives beyond linear systems, we ran a deliberately small two-layer network experiment. The model has architecture $3 \to 3 \to 1$ with 9 first-layer weights and 3 last-layer weights. We test two SiLU surrogates: the quadratic truncation $\varphi_2(t) = 2^{-1}t + 2^{-2}t^2$ and the quartic Taylor polynomial $\varphi_4(t) = \varphi_2(t) - 48^{-1}t^4$. After clearing denominators, the quadratic rows in Table 1 enumerate all 12 parameters over $\mathbb{F}_p$. The quartic rows instead use a capped search over the 9 first-layer weights and solve the 3 last-layer weights from the induced linear subsystem when possible. Every completed candidate is checked against the full stationarity system and the rank of the 12×12 Jacobian.

It shows that root existence modulo p is not the only obstacle: the search found many modular roots, but they lay on singular loci where the standard square-system Hensel condition fails. Thus, for nonlinear networks, the initial-root problem should be studied together with the Jacobian-rank distribution of the roots found by the search.

Across both experiments we distinguish four pipeline failure points: (a) no nonsingular seed, i.e. $\det J \equiv 0 \pmod{p}$, (b) no solution to the Hensel correction equation, (c) failed low-height reconstruction, and (d) failed verification in the original finite formulation.

5. Related Work

Low-bit neural-network work focuses on quantization formats, calibration, and training recipes [1–8]. We instead ask what exact algebraic stationarity problem remains after finite rational encoding and polynomial surrogates are fixed. Finite-precision arithmetic motivates rational encodings and polynomial approximations [9, 12], while Hensel lifting, rational reconstruction, and polynomial-system viewpoints are standard in number theory and computational algebra [13–17]. Our contribution is to connect these tools to a narrowly specified low-bit surrogate and expose initial-root and Jacobian nonsingularity bottlenecks.

6. Conclusion

Finite-bit polynomial surrogates lead to integer stationarity systems whose modular roots are candidates, not certificates. The experiments show that lift-reconstruct-verify works in linear regression, while the MLP seed search exposes the main nonlinear bottleneck: roots may exist modulo p but concentrate on singular Jacobian loci.

References

[1] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale," in *Advances in Neural Information Processing Systems*, 2022. [Online]. Available: https://arxiv.org/abs/2208.07339

[2] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers," in *International Conference on Learning Representations*, 2023. [Online]. Available: https://arxiv.org/abs/2210.17323

[3] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han, "SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models," in *International Conference on Machine Learning*, 2023. [Online]. Available: https://arxiv.org/abs/2211.10438

[4] J. Lin *et al.*, "AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration," in *Proceedings of Machine Learning and Systems*, 2024. [Online]. Available: https://arxiv.org/abs/2306.00978

[5] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," in *Advances in Neural Information Processing Systems*, 2023. [Online]. Available: https://arxiv.org/abs/2305.14314

[6] P. Micikevicius *et al.*, "FP8 Formats for Deep Learning," *arXiv preprint arXiv:2209.05433*, 2022, [Online]. Available: https://arxiv.org/abs/2209.05433

[7] H. Xi, C. Li, J. Chen, and J. Zhu, "Training Transformers with 4-bit Integers," in *Advances in Neural Information Processing Systems*, 2023. [Online]. Available: https://arxiv.org/abs/2306.11987

[8] M. Courbariaux, Y. Bengio, and J.-P. David, "BinaryConnect: Training Deep Neural Networks with Binary Weights During Propagations," in *Advances in Neural Information Processing Systems*, 2015. [Online]. Available: https://arxiv.org/abs/1511.00363

[9] "IEEE Standard for Floating-Point Arithmetic," *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, no. , pp. 1–84, 2019, doi: 10.1109/IEEESTD.2019.8766229.

[10] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011, [Online]. Available: https://www.jmlr.org/papers/v12/pedregosa11a.html

[11] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least Angle Regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004, doi: 10.1214/009053604000000067.

[12] J.-M. Muller *et al.*, *Handbook of Floating-Point Arithmetic*, 2nd ed. Birkhauser, 2018.

[13] F. Q. Gouvea, *p-adic Numbers: An Introduction*, Third edition. in Universitext. Springer, 2020. Accessed: Dec. 21, 2025. [Online]. Available: https://www.amazon.com/p-adic-Numbers-Fernando-Quadros-Gouvea/dp/3540629114

[14] N. Koblitz, *p-adic Numbers, p-adic Analysis, and Zeta-Functions*, 2nd ed. Springer, 1984.

[15] J.-P. Serre, *Local Fields*. Springer, 1979.

[16] J. von zur Garthen and J. Gerhard, *Modern Computer Algebra*. 1999.

[17] D. A. Cox, J. Little, and D. O'Shea, *Ideals, Varieties, and Algorithms*, 4th ed. Springer, 2015.

[18] A. G. Khovanskii, *Fewnomials*. American Mathematical Society, 1991.

[19] B. Sturmfels, *Solving Systems of Polynomial Equations*. American Mathematical Society, 2002.

Appendix A. Background

This section collects the algebraic and numerical background used later in the paper. We first recall the modular arithmetic needed for Hensel lifting in Appendix A.1. We then record the finite-bit rational encoding and finite-precision function-approximation assumptions used to build polynomial systems in Appendix A.2 and Appendix A.3.

A.1. Modular Arithmetic

The central object of this work is a p-adic number, that is, an element of the field $\mathbb{Q}_p$. The field $\mathbb{Q}_p$ is usually introduced as the completion of the rational numbers $\mathbb{Q}$ with respect to the p-adic absolute value. For our purposes, the digit expansion is more useful: a p-adic number is represented as an infinite series in powers of p with coefficients in $\{0, 1, ..., p-1\}$.

Definition A.1.1 *(p-adic number)* Let p be a prime. The field of p-adic numbers is the set of formal series

$$\mathbb{Q}_p = \left\{ \sum_{n=N}^{\infty} a_n p^n \mid N \in \mathbb{Z}, a_n \in \{0, 1, ..., p-1\} \right\} \tag{8}$$

with the convention that every nonzero element has a unique expression

$$x = \sum_{n=N}^{\infty} a_n p^n, \quad a_N \neq 0, \quad a_n \in \{0, 1, ..., p-1\}. \tag{9}$$

The subring of p-adic integers is

$$\mathbb{Z}_p = \left\{ \sum_{n=0}^{\infty} a_n p^n \mid a_n \in \{0, 1, ..., p-1\} \right\}, \quad \mathbb{Q}_p = \bigcup_{N \in \mathbb{Z}} p^N \mathbb{Z}_p. \tag{10}$$

For nonzero x, the smallest exponent N in this expansion is called the p-adic valuation and is denoted by $v_p(x) = N$. For $x = 0$, all digits are zero and one sets $v_p(0) = \infty$. The p-adic absolute value is

$$|x|_p = p^{-v_p(x)}. \tag{11}$$

The geometric and topological properties of $\mathbb{Q}_p$ differ substantially from those of $\mathbb{Q}$ or $\mathbb{R}$. The metric on $\mathbb{Q}_p$ is defined by

$$d_p(x, y) = |x - y|_p, \tag{12}$$

so two numbers are close when their difference is highly divisible by p. Instead of only satisfying the usual triangle inequality

$$d(x, z) \leq d(x, y) + d(y, z) \tag{13}$$

the p-adic metric satisfies the stronger ultrametric inequality

$$d_p(x, z) \leq \max\{d_p(x, y), d_p(y, z)\} \tag{14}$$

for all $x, y, z \in \mathbb{Q}_p$. This changes the behavior of balls and limits. In particular, real geometric intuition about descent directions and local extrema does not transfer directly to p-adic spaces.

Derivatives still make sense for many p-adic functions, including polynomials. In this paper we only need the formal derivative on $\mathbb{Q}_p[X]$: it is the $\mathbb{Q}_p$-linear operation D determined by $D(X^k) = kX^{k-1}$ and extended by linearity. For polynomial functions, this formal derivative agrees with the usual difference-quotient derivative in the p-adic topology.

Hensel's lifting. Several closely related results in number theory are named after Kurt Hensel. We use the univariate version first, written in a Newton-like form. For references and further generalizations, we refer the reader to [13, 14] and Serre's *Local Fields* [15].

n	α_n	$f(\alpha_n)$	α_{n+1}	5-adic expansion
1	4	$6 \cdot 5^1$	14	$4 + 2 \cdot 5$
2	14	$16 \cdot 5^2$	64	$4 + 2 \cdot 5 + 2 \cdot 5^2$
3	64	$66 \cdot 5^3$	314	$4 + 2 \cdot 5 + 2 \cdot 5^2 + 2 \cdot 5^3$

Table 2: Tabulated Hensel's iterations for toy problem in Example A.1.5.

Theorem A.1.2 *(Hensel's lemma)* Let $F(X) = \sum_{k=0}^{n} a_k X^k$ be a polynomial with coefficients in $\mathbb{Z}_p$. Suppose that there exists $\alpha_1 \in \mathbb{Z}_p$ such that $|F(\alpha_1)|_p < 1$ and $|F'(\alpha_1)|_p = 1$; equivalently, $F(\alpha_1) \equiv 0 \pmod{p}$ and $F'(\alpha_1) \not\equiv 0 \pmod{p}$. Then the sequence

$$\alpha_{n+1} = \alpha_n - \frac{F(\alpha_n)}{F'(\alpha_n)}, \quad n \geq 1 \tag{15}$$

is well-defined in $\mathbb{Z}_p$ and converges to the unique $\alpha \in \mathbb{Z}_p$ such that $|\alpha - \alpha_1|_p < 1$ and $F(\alpha) = 0$.

For computation, Theorem A.1.2 can be implemented as a compatible sequence of residues. Given a simple root $r_1 \in \mathbb{Z}/p\mathbb{Z}$, one constructs $r_N \in \mathbb{Z}/p^N\mathbb{Z}$ satisfying $F(r_N) \equiv 0 \pmod{p^N}$. In the usual digit-by-digit form, if r_k is known modulo p^k, the next digit $t_k \in \mathbb{Z}/p\mathbb{Z}$ is obtained from the linear congruence

$$F'(r_k)t_k \equiv -F(r_k)/p^k \pmod{p}, \quad r_{k+1} = r_k + p^k t_k. \tag{16}$$

The congruence has a unique solution precisely because $F'(r_k)$ is a unit modulo p.

For a square system $F : \mathbb{Z}_p^d \to \mathbb{Z}_p^d$, the analogous nonsingularity condition is that the Jacobian matrix $J_F(r_1)$ is invertible modulo p, equivalently $\det J_F(r_1) \not\equiv 0 (\mathrm{mod}\, p)$. If this condition fails, Hensel's lemma does not give a unique lift: a seed may have no lift, one lift, several isolated lifts, or positive-dimensional families of lifts.

Rational reconstruction. Hensel lifting often produces increasingly accurate residues modulo p^N. If the intended object is a rational number $a/b \in \mathbb{Q}$ with $\gcd(b, p) = 1$, then it has a well-defined residue modulo p^N. The inverse problem of recovering a small rational number from its residue is called rational reconstruction (see [16] for a computer algebra reference).

Definition A.1.3 *(Rational reconstruction problem)* Given a residue $x \pmod{m}$, find $(a, b) \in \mathbb{Z}^2$ such that $\gcd(a, b) = 1$, $b > 0$, $\gcd(b, m) = 1$, $|a| \leq A$, $b \leq B$, and $a \equiv bx \pmod{m}$.

In other words, rational reconstruction in Definition A.1.3 means finding two integers $a, b \in \mathbb{Z}$ from $x \pmod{m}$ such that

$$\frac{a}{b} \equiv x \pmod{m}. \tag{17}$$

Uniqueness of such a pair of integers depends on the absolute values a and b and is given by the following theorem.

Theorem A.1.4 *(Uniqueness of rational reconstruction)* If $2AB < m$ then there is at most one rational number $\frac{a}{b}$ reducing to a given residue $x \pmod{m}$.

The Example A.1.5 demonstrates step-by-step recovery of rational roots for a quadratic univariate polynomial. The calculation starts by finding initial roots modulo p, then lifts each root, and finally uses rational reconstruction to recover rational roots.

Example A.1.5 *(Hensel lifting for a quadratic equation)* Consider $f(x) = 2x^2 + x - 6$, which factors as $(2x - 3)(x + 2)$ over $\mathbb{Q}$ with roots $x = \frac{3}{2}$ and $x = -2$. We find both roots 5-adically, starting from trivial residues.

1. **Trivial initial roots (mod 5).** Brute-force evaluation over $\mathbb{F}_5$

$$f(0) \equiv 4, \quad f(1) \equiv 2, \quad f(2) \equiv 4, \quad f(3) \equiv 0, \quad f(4) \equiv 0 \pmod 5 \tag{18}$$

yields two initial residues $\alpha_1 \in \{3, 4\}$. Each is a single digit – a coarse approximation that does not yet reveal the true roots (e.g. $f(4) \equiv 0 \pmod 5$ but $f(4) = 30$).

2. **Newton-Hensel lifting (from $\alpha_1 = 4$).** The digit-by-digit Newton-Hensel update accumulates one 5-adic digit at each step. In this example $v_5(f(\alpha_n)) = n$ for the displayed residues. The sequence $4, 14, 64, 314, \ldots$ is the 5-adic expansion of $\frac{3}{2} = 4 + 2 \cdot 5 + 2 \cdot 5^2 + 2 \cdot 5^3 + \ldots$ (see Table 2). Full Newton iteration over $\mathbb{Q}_5$ can increase the valuation faster, but the table reports the one-digit lifting form.

3. **Rational reconstruction.** After just two lifting steps, the residue $\alpha_2 = 14 \pmod{25}$ already encodes the full answer: $2 \cdot 14 = 28 \equiv 3 \pmod{25}$, so the extended Euclidean algorithm recovers $\alpha_2 \equiv 3/2 \pmod{25}$. The other branch lifts $\alpha_1 = 3$ to $23 \equiv -2 \pmod{25}$, recovering the integer root.

A.2. Floating-Point Arithmetic

The connection to IEEE-754 arithmetic [9] is representational, not algebraic. Once a finite format and scaling convention are fixed, each stored number denotes a rational value. For a normalized binary floating-point value, one may write

$$x = \pm 2^e \left(1 + \sum_{k=1}^{M} \frac{m_k}{2^k}\right) = \pm \sum_{k=e-M}^{e} m_{k-e} 2^k. \tag{19}$$

This identity is only an encoding statement: it does not make floating-point addition, multiplication, rounding, overflow, or subnormal behavior isomorphic to arithmetic in $\mathbb{Q}_p$. In this paper the finite-bit assumption is used only to obtain rational coefficients and integer variables after denominator clearing. The modular systems studied below are therefore models of the integerized stationarity equations, not simulations of IEEE-754 execution.

A.3. Function Approximation

Non-polynomial activations and losses enter the modular formulation through explicit polynomial surrogates. Each surrogate must specify its interval or partition, degree, rationalized coefficients, and controlled error. In our experiments the surrogate is part of the problem definition; we verify only the resulting integer stationarity equations. This mirrors the use of polynomial approximations in finite-precision function implementations, but without modeling a particular library routine.

Appendix B. Proofs

Proposition B.6 *(One-way modular reduction)* With the assumptions 2.1 and 2.2, any bounded finite-bit code vector that satisfies the integer stationarity equations for the encoded surrogate of (1) gives a residue class satisfying (2). A solution of (2) need not be a stationary point or a local minimizer of the original finite problem.

Proof sketch. After fixing polynomial pieces and the finite representation, all coefficients in F are rational functions of fixed scales and integer code variables. Multiplication by the common denominator D gives $\tilde{F} \in \mathbb{Z}[\boldsymbol{w}]$ without changing the rational zero set before reduction. Any exact finite-bit stationary code vector is integral and therefore has a residue modulo p^N satisfying (2). If

one instead works directly with rational value variables, their denominators must be units modulo p; otherwise the scale must first be absorbed into integer codes. The reverse implication fails because a residue class can satisfy the congruence without reconstructing to a feasible finite-bit point, and stationarity itself does not certify finite-domain local optimality.

Modular solutions should therefore be interpreted as candidates. Their training loss, finite-domain local optimality, and generalization behavior are empirical properties to be measured after reconstruction and verification.

Appendix C. Extended Discussion of Limitations

C.1. Seed Search

Hensel lifting starts from a root modulo p, but finding such roots is a hard finite-field search problem for neural networks. Exhaustive search scales as p^d in the number of parameters, and continuous training followed by quantization does not automatically produce an exact modular root. Initial-root discovery should therefore be treated as a central research problem rather than an implementation detail.

Sparse polynomial structure does not remove this limitation by itself. For some polynomial networks, fewnomial, Newton-polytope, or Pfaffian viewpoints may give useful bounds on the number of non-degenerate solutions or describe how the root set depends on width and depth [18, 19]. Such bounds are not search algorithms. Even if the number of useful seeds were polynomial in the number of parameters, random search over $\mathbb{F}_p^d$ would still have success probability on the order of a polynomial divided by p^d. These estimates are therefore most useful for diagnosing uniqueness, degeneracy, and root geometry, not for claiming that modular seed search is scalable.

A useful next step is to measure the geometry of the root set, not only the presence or absence of roots. Rank histograms of modular roots, Newton-polytope and monomial-count diagnostics, and fewnomial-style estimates may help distinguish two cases: full-rank roots exist but are rare, or the searched roots concentrate on a singular variety. These cases require different algorithms.

The most plausible scalable route is therefore correction rather than blind enumeration: start from a real or fixed-point trained candidate, reduce it modulo p, and take a Newton correction step over $\mathbb{F}_p$ when the lifted Jacobian is invertible. The correction step itself is standard finite-field linear algebra; the open question is whether the relevant determinant is nonzero modulo useful small primes for trained low-bit models. Equivalently, one needs empirical and theoretical control of the bad primes dividing the determinant of the corrected or deflated Jacobian.

C.2. Lifting Conditions

Ordinary square-system Hensel lifting requires the Jacobian of the integer polynomial system to have full rank modulo p at the seed. If this condition fails, a root modulo p may have no lift, multiple lifts, or a positive-dimensional family of lifts. The small polynomial MLP experiment in Section 4.2 illustrates this problem: many raw modular roots were found, but all observed roots were singular and therefore unusable as ordinary Hensel seeds.

The choice of prime and polynomial approximation is another limitation. Clearing rational denominators can introduce bad reductions at primes dividing the clearing factor. Piecewise polynomial approximations also require a specified approximation interval, degree, and error criterion; changing these choices changes the modular system. The experiments in this paper only test small fixed systems and should not be read as evidence that the approach scales to modern networks.

There is an additional prime-dependent issue for polynomial activations. If an activation degree d is divisible by the chosen prime p, Frobenius identities can collapse the finite-field model. For

p	Nonsingular Seeds	Verified Solves	Success Rate
2	0/100	0/100	0.00
3	67/100	67/100	0.67
5	83/100	83/100	0.83
7	82/100	82/100	0.82
11	90/100	90/100	0.90
13	95/100	95/100	0.95

Table 3: Synthetic linear-regression outcomes for $d = 32$ and 100 random seeds. In this linear setting, each nonsingular initial system produced a verified rational solution of the exact normal equations within the precision budget.

example, powers may linearize over $\mathbb{F}_p$, and gradient blocks containing a factor d may vanish identically modulo p. This can be useful structure, but it also makes the ordinary Jacobian singular. In such cases the correct system may require deflation, for example replacing a vanished block $F_A = dG_A$ by G_A before applying a Newton or Hensel step. The present experiments use the standard nonsingular square-system condition and do not analyze deflated or singular Hensel variants.

Appendix D. Experiment Details

All reported runs use exact integer or finite-field arithmetic on CPU. The linear-regression sweep reports 100 independent random draws for each prime. The MLP rows report the number of residues visited explicitly; capped rows should be read as search traces, not as exhaustive absence claims.

D.1. Linear Regression on Diabetes Dataset

We also used the `scikit-learn` Diabetes dataset [10, 11] as a real-data sanity check for the same fixed-point normal-equation solver. In local runs, the lifted residues were reconstructed and verified after roughly 320–350 Hensel steps. We do not use this as the main empirical evidence because the synthetic sweep above gives repeated trials with controlled dimensions and verification statistics.