Degree Filtration and Domain-Tower FFTs over M31 via the Boolean Zeta Transform and Kernel Polynomials

Ali Mkhida
Algorizk Labs, Bordeaux, France
ali.mkhida@algorizk.xyz
ORCID: 0009-0009-2101-9070

Abstract

We study Boolean kernel polynomials for degree-bounded univariate polynomials built from arbitrary monic coordinates of degrees $1, 2, 4, \ldots$. The coefficient map from the kernel polynomials to the associated degree-ordered product basis is a complemented Boolean zeta transform. Our main theorem gives an exact description of the ordinary degree bound $\deg U < 2^k$ directly in the full kernel coefficient array: after the low and high bits of the Boolean index are separated, the coefficients for fixed low bits follow an alternating-sign pattern determined by the high bits. Equivalently, the degree-$< 2^k$ projector is a tensor product of local one-dimensional projectors. In characteristic two the signs disappear, so the coefficients are simply constant when the low bits are fixed and the high bits vary.

When the polynomial coordinates are generated by a quadratic map, the same representation gives an $O(N \log N)$ recursive transform on evaluation sets organized in pairs of preimages. Over the Mersenne prime $p = 2^{31} - 1$, the rescaling $s = 2c$ turns $s \mapsto s^2 - 2$ exactly into the one-variable map $c \mapsto 2c^2 - 1$ used by the recursive stages of Circle FFTs. A construction from the norm-one subgroup supplies 29 levels in which every point has two distinct preimages. The optimized Rust/ARM64 implementation agrees exactly with a scalar reference for the quadratic recursion on 543 test vectors. Seven-round measurements at $N = 2^{18}, 2^{19}, 2^{20}$ give transform-kernel running times in the same range as Stwo's SIMD Circle FFT under the stated measurement boundary; at $N = 2^{20}$ the median per-round kernel/Stwo ratio is 0.979510.

Keywords: Boolean zeta transform; kernel polynomials; degree filtration; domain-tower FFT; Mersenne-31; Circle FFT; finite-field polynomial evaluation.

2020 MSC: 12Y05, 11T06, 68W30.

Paper guide.

Related work · Kernel coordinates · Degree-bound theorem · Recursive evaluator · M31/Circle tower
Rust/ARM64 implementation · Benchmarks · Limitations · Future work · Conclusion · Notation appendix

Contents

1 Introduction

Fast polynomial evaluation depends not only on an evaluation domain but also on the coordinates in which the polynomial is represented. In monomial coordinates, ordinary degree is immediate, whereas recursive evaluation structure can be difficult to expose. Alternative finite-field bases reverse this balance by adapting the coordinates to a recursive domain or tower. Additive FFTs based on subspace polynomials [5], the Lin–Chung–Han family and its subsequent developments [12, 11, 4], and more general recursive product bases [9] exemplify this basis-aware approach. A separate line of work obtains fast polynomial algorithms from elliptic-curve or related degree-two geometries [1, 2, 6], while Circle-based Reed–Solomon codes and Circle STARKs exploit the norm-one circle and its quadratic one-variable recursion [8, 7, 3].

This paper studies a Boolean kernel coordinate system attached to an arbitrary monic degree-doubling sequence. Let $\mathbb{F}$ be a field and let

$$P_i(X) \in \mathbb{F}[X], \qquad P_i \text{ monic}, \qquad \deg P_i = 2^i, \qquad 0 \le i < m.$$

For $a, y \in \{0,1\}^m$, define

$$B_a(X) = \prod_{i=0}^{m-1} P_i(X)^{a_i}, \qquad K_y^P(X) = \prod_{i=0}^{m-1} (P_i(X) + y_i).$$

The product family $(B_a)_a$ is degree ordered: B_a is monic of ordinary degree

$$|a|_2 = \sum_{i=0}^{m-1} a_i 2^i,$$

and consequently forms a basis of $\mathbb{F}[X]_{<2^m}$. Expanding the kernel family in that product basis gives the exact coordinate relation

$$u_a = \sum_{y \ge \bar{a}} \lambda_y, \tag{1}$$

when

$$U = \sum_y \lambda_y K_y^P = \sum_a u_a B_a.$$

Thus the kernel-to-product change of coordinates is the complemented Boolean zeta transform, with inverse given by Boolean Möbius inversion [14]. The zeta/Möbius machinery itself is classical; its role here is to expose the interaction between kernel coordinates and ordinary polynomial degree.

The central structural result describes the ordinary degree bound $\deg U < 2^k$ directly in the full kernel coefficients. For $0 \le k \le m$, split $y = (x, h)$ with $x \in \{0,1\}^k$ containing the low bits and $h \in \{0,1\}^{m-k}$ containing the high bits. Then

$$\deg U < 2^k \quad \Longleftrightarrow \quad \lambda_{(x,h)} = (-1)^{m-k-\mathrm{wt}(h)} \mu_x \tag{2}$$

for a unique family $(\mu_x)_x$. Equivalently, projection onto $\mathbb{F}[X]_{<2^k}$ is a tensor product of local one-dimensional projectors in kernel coordinates. The corresponding alternating-sign sum over the high bits is

$$\sum_{h \in \{0,1\}^{m-k}} (-1)^{m-k-\mathrm{wt}(h)} K^P_{(x,h)}(X) = \prod_{i=0}^{k-1} (P_i(X) + x_i). \tag{3}$$

In characteristic two, the signs disappear: for fixed low bits x, the coefficients are constant as the high bits h vary. These statements hold for the full arbitrary-monic degree-doubling family above, including choices of P_i beyond the monomials X^{2^i}.

This characterization has a direct algorithmic consequence. To test whether a kernel coefficient array represents a polynomial of degree below 2^k, one checks the alternating-sign rule independently for each fixed low-bit index. The corresponding projector applies the same one-dimensional local map on each of the $m - k$ high coordinates. The degree restriction can therefore be enforced directly in kernel coordinates, without first converting the whole array to the product basis.

Fast evaluation requires additional structure beyond monicity and degree doubling. We therefore specialize to a quadratic tower generated by

$$\phi(X) = X^2 - \delta, \qquad P_0(X) = X, \qquad P_{i+1}(X) = P_i(\phi(X)).$$

Splitting the least significant Boolean coordinate gives a one-step recursive decomposition

$$U(X) = X\, A(\phi(X)) + (X+1) B(\phi(X)) = D(\phi(X)) + X\, C(\phi(X)), \tag{4}$$

with $C = A + B$ and $D = B$. For the paired preimages u and $-u$, with $\phi(u) = \phi(-u) = t$, this becomes

$$\begin{pmatrix} U(u) \\ U(-u) \end{pmatrix} = \begin{pmatrix} 1 & u \\ 1 & -u \end{pmatrix} \begin{pmatrix} D(t) \\ C(t) \end{pmatrix}. \tag{5}$$

When the evaluation domains form a complete two-to-one hierarchy under ϕ, this recursion yields an $O(N \log N)$ domain-tower transform.

The concrete systems specialization uses the Mersenne prime

$$p = 2^{31} - 1$$

and the affine map

$$\phi(s) = s^2 - 2.$$

Writing

$$\pi(c) = 2c^2 - 1,$$

the rescaling $s = 2c$ satisfies

$$\phi(2c) = 2\pi(c). \tag{6}$$

Consequently,

$$P_i(2c) = 2\,\pi^{\circ i}(c),$$

so the degree-doubling product basis becomes, up to a diagonal scaling, the one-variable product basis used by the recursive Circle-FFT stages. The exact identity concerns these recursive one-variable stages; a complete Circle FFT also includes its circle-to-line step, point ordering, evaluation subsets, stage multipliers, bit reversal, and memory layout.

For M31, the norm-one subgroup of $\mathbb{F}_{p^2}^{\times}$ has order $p + 1 = 2^{31}$. From an element of exact order 2^{31} one obtains a complete paired-preimage hierarchy for $s \mapsto s^2 - 2$. The base domain has size 2^{29}, and the construction gives 29 successive two-to-one maps with two distinct preimages at every level. At the next level the two preimages merge at the critical point. This supplies exactly the domain tower required by the recursive transform.

The implementation section treats the systems result separately from the algebraic contribution. The optimized Rust/ARM64 path is verified against a scalar reference for the quadratic recursion on the recorded test set and then compared with Stwo's public SIMD Circle-FFT entry point [15]. The benchmark uses

$$\log_2 N \in \{18, 19, 20\},$$

seven independent rounds, and nine repetitions per round. At $\log_2 N = 20$, the aggregate kernel and Stwo medians are respectively 6.924750 ms and 7.058000 ms, while the median of the per-round kernel/Stwo ratios is 0.979510 with observed range $[0.976630, 0.996149]$. Across the tested sizes, the measurements show stable, similar transform-kernel running times on the tested Apple ARM64 configuration. End-to-end prover timing is outside this benchmark.

Contributions. The paper makes four contributions.

1. It formulates the Boolean kernel basis for arbitrary monic degree-doubling coordinates and records its exact complemented zeta/Möbius relation to the corresponding degree-ordered product basis.

2. It characterizes the ordinary degree bound $\deg U < 2^k$ directly in the full kernel coefficient array, proves the associated alternating-sign collapse identity, and gives the equivalent tensor product of local one-dimensional projectors. In characteristic two, coefficients are constant when the low bits are fixed and the high bits vary.

3. It derives the recursive quadratic evaluator, instantiates its complete paired-preimage domain hierarchy over M31, and proves its exact diagonal-scaling relation to the one-variable recursion used by recursive Circle FFTs.

4. It validates the resulting M31 transform in an optimized Rust/ARM64 implementation and reports a reproducible transform-kernel comparison with the recorded Stwo SIMD baseline.

Scope. The general algebraic results concern Boolean zeta/kernel coordinates and ordinary polynomial degree over arbitrary fields. The $O(N \log N)$ transform requires the additional quadratic domain-tower structure. The M31 section identifies the exact rescaling to the one-variable recursion used by Circle FFTs. The implementation measurements concern the native transform kernel on the recorded Apple ARM64 environment; domain and stage-multiplier construction, benchmark setup, warmup, and output interpretation are outside the timed region.

Organization. Section 2 positions the results against additive and algebraic FFT constructions and places the results in the existing FFT and finite-field literature. Section 3 develops the degree-doubling product and kernel coordinates. Section 4 proves the coefficient characterization of degree below 2^k and its projector. Section 5 derives the recursive quadratic evaluator. Section 6 specializes the construction to M31, proves the exact Circle-FFT rescaling, and constructs the complete evaluation-domain tower. Section 7 gives the correctness and measurement boundary, Section 8 reports the benchmark data, Section 9 discusses the scope and limitations, and Section 11 concludes.

2 Related work

The results in this paper connect three established areas: polynomial bases adapted to recursive evaluation, Boolean zeta/Möbius transforms, and degree-two FFT constructions.

Additive FFTs and polynomial bases. Over characteristic-two fields, Lin, Chung, and Han introduced a polynomial basis adapted to additive FFT structure and Reed–Solomon encoding [12]; subsequent work developed the basis and its algorithms further [11]. Coxon gave fast evaluation, interpolation, and conversion algorithms for this family, including conversions among the Lin–Chung–Han, Newton, Lagrange, and monomial representations [4]. More generally, Li and Xing study recursive product bases in a function-field framework [9]. These works show how degree-ordered coordinates can expose recursive polynomial algorithms.

The product family used here,

$$B_a = \prod_i P_i^{a_i}, \qquad \deg P_i = 2^i,$$

uses the same degree-ordering principle. The additional structure studied in this paper comes from the full kernel family

$$K_y^P = \prod_i (P_i + y_i).$$

Its coefficient map to the product basis is a complemented Boolean zeta transform. The main degree theorem then asks a different question: which full kernel coefficient arrays represent polynomials satisfying $\deg U < 2^k$? The answer is the alternating-sign rule proved in Section 4. Boolean zeta/Möbius inversion itself is the classical Boolean-lattice transform of Rota [14].

Algebraic FFTs over more general fields. The elliptic-curve FFT (ECFFT) replaces roots of unity by evaluation sets derived from elliptic-curve groups and uses chains of degree-two isogenies to obtain FFT-style divide-and-conquer algorithms over general finite fields [1, 2]. Li and Xing give a broader function-field treatment that contains multiplicative and additive FFTs and also covers the case in which $q+1$ is smooth [9]. Haböck's G-FFT gives another degree-two construction associated with group doubling [6]. These works address the geometry and recursive structure needed for fast polynomial evaluation when the multiplicative group of the field does not provide a convenient smooth subgroup.

Sections 3 and 4 are independent of that domain-construction problem: they require only monicity and $\deg P_i = 2^i$. The domain geometry enters in Section 5, where the coordinates are generated by a quadratic map and the evaluation sets are organized as successive pairs of preimages.

Circle-domain transforms. Reed–Solomon codes over the norm-one circle exploit Mersenne fields and the circle geometry to obtain efficient encoding domains [8]. Circle STARKs use the circle-to-line projection followed by the quadratic x-coordinate recursion in their FFT and proximity machinery [7]. S-two implements this architecture over M31 in a complete STARK system [3, 15].

For the M31 specialization in this paper, the exact rescaling

$$s = 2c, \qquad s^2 - 2 = 2(2c^2 - 1)$$

identifies the quadratic map used by the kernel transform with the one-variable map used by the recursive Circle-FFT stages. The corresponding product-basis evaluation matrices differ by a diagonal scaling, and the kernel coefficients add the fixed Boolean-zeta change of coordinates. A complete Circle FFT also contains its circle-to-line step, point ordering, evaluation subsets, stage multipliers, and implementation layout. The comparison in the implementation section is therefore made at the common recursive-transform boundary.

Positioning of the present result. The algebraic contribution is the exact description of the ordinary power-of-two degree bounds in full kernel coefficients, together with the alternating-sign collapse identity and the local tensor projector. The quadratic and M31 sections turn that coefficient theorem into a concrete domain-tower transform, and the Rust/ARM64 section measures the resulting transform at the same software boundary as the selected Stwo Circle-FFT baseline.

3 The kernel framework

This section fixes the polynomial and Boolean-coordinate conventions used in the remainder of the paper. The construction is valid over an arbitrary field and depends only on monicity and degree doubling. We first record the degree-ordered product basis and then show explicitly how the Boolean zeta transform lifts its columns to the kernel polynomials used throughout the paper.

3.1 Degree-doubling product coordinates

Let $\mathbb{F}$ be a field, let $m \geq 0$, and put

$$N = 2^m, \qquad \mathbb{B}_m = \{0,1\}^m.$$

The symbols 0 and 1 in $\mathbb{B}_m$ are combinatorial bits; in polynomial expressions they denote their images in $\mathbb{F}$. For $a = (a_0, \ldots, a_{m-1}) \in \mathbb{B}_m$, define

$$|a|_2 = \sum_{i=0}^{m-1} a_i 2^i, \qquad \mathrm{wt}(a) = \sum_{i=0}^{m-1} a_i, \qquad \bar{a} = (1 - a_0, \ldots, 1 - a_{m-1}).$$

We write $a \leq y$ for the componentwise order. The binary-value map $a \mapsto |a|_2$ is a bijection from $\mathbb{B}_m$ to $\{0, \ldots, N-1\}$. Boolean vectors are ordered below by increasing binary value, with coordinate zero the least significant bit. When $m = 0$, $\mathbb{B}_0$ consists of the empty tuple, and all sums and products over the empty coordinate set have their usual values zero and one.

Fix a sequence

$$P_0, \ldots, P_{m-1} \in \mathbb{F}[X], \qquad P_i \text{ monic}, \qquad \deg P_i = 2^i. \tag{7}$$

For $a \in \mathbb{B}_m$, define the coordinate product

$$B_a(X) = \prod_{i=0}^{m-1} P_i(X)^{a_i}. \tag{8}$$

Let $\mathbb{F}[X]_{<d}$ denote the $\mathbb{F}$-vector space of polynomials of ordinary degree strictly less than d. We use the convention $\deg 0 = -\infty$.

Theorem 3.1 (Degree ordering and unitriangularity)**.** *Under* (7)*, every B_a is monic and*

$$\deg B_a = |a|_2. \tag{9}$$

Consequently, $(B_a)_{a \in \mathbb{B}_m}$, ordered by increasing $|a|_2$, is a basis of $\mathbb{F}[X]_{<N}$. More precisely, if $a(n) \in \mathbb{B}_m$ is determined by $|a(n)|_2 = n$ and

$$T_{n,j} = [X^j] B_{a(n)}, \qquad 0 \leq n, j < N,$$

then T is lower unitriangular. Equivalently, the matrix whose columns are the B_a in the ordered monomial basis is upper unitriangular.

Proof. For each a, degrees add and leading coefficients multiply, so

$$\deg B_a = \sum_{i=0}^{m-1} a_i \deg P_i = \sum_{i=0}^{m-1} a_i 2^i = |a|_2, \qquad \mathrm{lc}(B_a) = 1.$$

Thus $B_{a(n)} = X^n + \sum_{j<n} T_{n,j} X^j$. This is exactly lower unitriangularity of T. Since a unitriangular matrix is invertible over every field, the N products form a basis of the N-dimensional space $\mathbb{F}[X]_{<N}$. For $m = 0$, this reduces to the basis $(B_\varnothing) = (1)$ of $\mathbb{F}[X]_{<1}$. □

The same argument also identifies each ordinary-degree initial segment: for $0 \le d \le N$,

$$\mathbb{F}[X]_{<d} = \operatorname{span}_{\mathbb{F}}\{B_a : |a|_2 < d\}. \tag{10}$$

Indeed, the corresponding leading principal submatrix of T is again lower unitriangular.

3.2 Boolean kernels

For $y = (y_0, \dots, y_{m-1}) \in \mathbb{B}_m$, define

$$K_y^P(X) = \prod_{i=0}^{m-1} (P_i(X) + y_i). \tag{11}$$

Each factor is monic of degree 2^i; hence every K_y^P is monic of degree $\sum_i 2^i = N - 1$ and belongs to $\mathbb{F}[X]_{<N}$. In particular, unlike the products B_a, the kernel family is not ordered by the degrees of its individual elements.

Lemma 3.2 (Kernel expansion)**.** *For every $y \in \mathbb{B}_m$,*

$$K_y^P = \sum_{a \in \mathbb{B}_m} \left(\prod_{i : a_i = 0} y_i \right) B_a = \sum_{\substack{a \in \mathbb{B}_m \\ y \ge \bar{a}}} B_a. \tag{12}$$

Proof. Expanding (11), choose P_i from the ith factor when $a_i = 1$ and y_i when $a_i = 0$. The resulting term is $B_a \prod_{i:a_i=0} y_i$. Because every y_i is a bit, this scalar is one exactly when $y_i = 1$ for all i such that $a_i = 0$, which is equivalent to $y \ge \bar{a}$; otherwise it is zero. □

The coefficient vector of K_y^P in the product basis is $a \mapsto \mathbf{1}[\bar{y} \le a]$, which is exactly the Boolean-zeta column indexed by $\bar{y}$. Thus each K_y^P is the polynomial lift of a Boolean zeta column through the product basis. This is the sense in which we use the term *kernel polynomial* in this paper.

Theorem 3.3 (Boolean kernel basis and exact coordinate transform)**.** *Under (7), the family $(K_y^P)_{y \in \mathbb{B}_m}$ is a basis of $\mathbb{F}[X]_{<N}$. If a polynomial $U \in \mathbb{F}[X]_{<N}$ is written in the two bases as*

$$U = \sum_{y \in \mathbb{B}_m} \lambda_y K_y^P = \sum_{a \in \mathbb{B}_m} u_a B_a, \tag{13}$$

then its coordinates satisfy

$$u_a = \sum_{y \geq \bar{a}} \lambda_y \qquad (a \in \mathbb{B}_m). \tag{14}$$

Conversely,

$$\lambda_y = \sum_{c \geq y} (-1)^{\mathrm{wt}(c)-\mathrm{wt}(y)} u_{\bar{c}} \qquad (y \in \mathbb{B}_m), \tag{15}$$

where the integer signs are interpreted through the canonical map $\mathbb{Z} \to \mathbb{F}$*.*

Proof. Substituting (12) into the first expansion in (13) and collecting the coefficient of B_a gives (14). Define the upper Boolean zeta transform by

$$(\zeta\lambda)(c) = \sum_{y \geq c} \lambda_y.$$

Then (14) is the identity $u_a = (\zeta\lambda)(\bar{a})$: the change of coordinates is the upper zeta transform followed by complementation of the output index. Möbius inversion on the Boolean lattice gives

$$\lambda_y = \sum_{c \geq y} (-1)^{\mathrm{wt}(c)-\mathrm{wt}(y)} (\zeta\lambda)(c),$$

which becomes (15) after substituting $(\zeta\lambda)(c) = u_{\bar{c}}$ [14]. Thus the coordinate map is invertible. Since $(B_a)_a$ is a basis by Theorem 3.1, its inverse image $(K_y^P)_y$ is also a basis. In characteristic two the displayed signs all map to one, but the same zeta and Möbius maps remain mutually inverse. □

The transform matrix in the row convention of (14) is

$$Z_m(a, y) = \mathbf{1}[y \geq \bar{a}]. \tag{16}$$

For one coordinate, $Z_1 = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 1 \end{smallmatrix}\right)$; up to the fixed ordering of Boolean indices, Z_m is its m-fold Kronecker power. This gives the tensor interpretation of the kernel basis and makes explicit that the coordinate matrix is independent of the coefficients of the P_i. The polynomials represented by those coordinates, however, still depend on the chosen sequence P.

3.3 Fast coordinate conversion

Equation (14) can be evaluated by the standard in-place Boolean-zeta schedule. Initialize $g(c) = \lambda_c$ for $c \in \mathbb{B}_m$. For each coordinate $i = 0, \ldots, m-1$, perform

$$g(c) \leftarrow g(c) + g(c + e_i) \qquad \text{for every } c \text{ with } c_i = 0, \tag{17}$$

where e_i changes the ith bit from zero to one. After all m stages, $g(c) = \sum_{y \geq c} \lambda_y$, and hence $u_a = g(\bar{a})$. Each stage uses $N/2$ field additions, so the zeta phase uses $(N/2)\log_2 N$ field additions, followed by an $O(N)$ complement permutation.

For the inverse conversion, initialize $g(c) = u_{\bar{c}}$ and replace the addition in (17) by subtraction. After the same m stages, $g(y) = \lambda_y$. Thus either direction costs $O(N \log N)$ field operations. In characteristic two the forward and inverse butterflies have the same field operation because subtraction equals addition. These counts concern coordinate conversion only; consequences for later algorithms and implementations are deferred to the corresponding sections.

3.4 Connection with standard basis-conversion tools

Degree-indexed products of degree-2^i coordinates appear in the Lin–Chung–Han construction and its subsequent development [12, 11]. Coxon studies conversion among the LCH, Newton, Lagrange, and monomial representations [4], while related recursive product bases occur in the broader framework of Li and Xing [9]. Boolean zeta/Möbius inversion is the classical Boolean-lattice transform of Rota [14].

In the present notation these two familiar ingredients meet in a particularly simple way: the kernel coefficients are converted to the degree-ordered product coefficients by the fixed complemented Boolean zeta matrix (14), independent of the coefficients of the P_i. The next section uses this explicit map to characterize the ordinary degree bound $\deg U < 2^k$ directly in the kernel coefficients.

4 Degree bounds in kernel coordinates

The kernel coordinates of Section 3 are not ordered by the ordinary degrees of the individual kernels: every full kernel K_y^P has degree $N - 1$. Nevertheless, the ordinary degree bound $\deg U < 2^k$ has a simple description in these coordinates. For fixed low bits, the coefficients indexed by the remaining high bits follow a one-dimensional alternating-sign pattern. In characteristic two the signs disappear, so those coefficients are constant as the high bits vary.

Fix the hypotheses of Section 3. Let $0 \leq k \leq m$, put $r = m - k$, and split Boolean indices as

$$y = (x, h), \qquad x \in \mathbb{B}_k, \qquad h \in \mathbb{B}_r,$$

where x contains the k low bits and h the r high bits. Define the truncated kernel

$$K_x^{P,[k]}(X) = \prod_{i=0}^{k-1} (P_i(X) + x_i), \qquad x \in \mathbb{B}_k. \tag{18}$$

The empty product for $k = 0$ is one. Applying Theorem 3.3 to the prefix $P_0, \dots, P_{k-1}$ shows that $(K_x^{P,[k]})_{x \in \mathbb{B}_k}$ is a basis of $\mathbb{F}[X]_{<2^k}$.

4.1 Alternating-sign collapse over the high bits

The elementary identity behind the degree characterization is obtained by summing over the high bits with alternating signs.

Lemma 4.1 (Alternating-sign collapse)**.** *For every $x \in \mathbb{B}_k$,*

$$\sum_{h \in \mathbb{B}_r} (-1)^{r - \mathrm{wt}(h)} K_{(x,h)}^P(X) = K_x^{P,[k]}(X). \tag{19}$$

The integer signs are interpreted through the canonical map $\mathbb{Z} \to \mathbb{F}$.

Proof. Factor the low coordinates out of the sum. Since $(-1)^{r-\mathrm{wt}(h)} = \prod_{i=k}^{m-1} (-1)^{1-h_i}$, distributivity gives

$$\begin{aligned} \sum_{h \in \mathbb{B}_r} (-1)^{r-\mathrm{wt}(h)} K_{(x,h)}^P(X) &= K_x^{P,[k]}(X) \prod_{i=k}^{m-1} \left(\sum_{b \in \{0,1\}} (-1)^{1-b} (P_i(X) + b) \right) \\ &= K_x^{P,[k]}(X) \prod_{i=k}^{m-1} \left(-(P_i(X)) + (P_i(X) + 1) \right) \\ &= K_x^{P,[k]}(X). \end{aligned}$$

The same calculation is valid in characteristic two, where $-1 = 1$. □

4.2 Exact characterization of degree below 2^k

We can now read the ordinary-degree constraint directly from the full kernel coefficients.

Theorem 4.2 (Degree below 2^k is an alternating-sign high-bit condition)**.** *Let*

$$U(X) = \sum_{x \in \mathbb{B}_k} \sum_{h \in \mathbb{B}_r} \lambda_{(x,h)} K_{(x,h)}^P(X) \in \mathbb{F}[X]_{<N}. \tag{20}$$

Then the following are equivalent:

1. $\deg U < 2^k$;

2. *there is a unique family* $(\mu_x)_{x\in\mathbb{B}_k}$ *such that*

$$\lambda_{(x,h)} = (-1)^{r-\mathrm{wt}(h)}\mu_x \qquad (x \in \mathbb{B}_k,\ h \in \mathbb{B}_r); \tag{21}$$

3. *U has the unique truncated-kernel expansion*

$$U(X) = \sum_{x\in\mathbb{B}_k} \mu_x K_x^{P,[k]}(X). \tag{22}$$

Proof. If (21) holds, then Lemma 4.1 gives

$$U = \sum_{x\in\mathbb{B}_k} \mu_x \sum_{h\in\mathbb{B}_r} (-1)^{r-\mathrm{wt}(h)} K^P_{(x,h)} = \sum_{x\in\mathbb{B}_k} \mu_x K_x^{P,[k]}.$$

Every truncated kernel has degree at most $2^k - 1$, so $\deg U < 2^k$.

Conversely, suppose $\deg U < 2^k$. The truncated kernels form a basis of $\mathbb{F}[X]_{<2^k}$, so there is a unique family $(\mu_x)_x$ satisfying (22). Expanding each truncated kernel by Lemma 4.1 yields

$$U = \sum_{x\in\mathbb{B}_k} \sum_{h\in\mathbb{B}_r} (-1)^{r-\mathrm{wt}(h)} \mu_x K^P_{(x,h)}.$$

The full kernels form a basis of $\mathbb{F}[X]_{<N}$ by Theorem 3.3; uniqueness of full kernel coordinates therefore forces (21). The same argument proves uniqueness of $(\mu_x)_x$. □

Corollary 4.3 (Characteristic-two fiber constancy)**.** *If* $\mathrm{char}\,\mathbb{F} = 2$*, then*

$$\deg U < 2^k \quad\Longleftrightarrow\quad \lambda_{(x,h)} = \mu_x \quad \textit{for all } x \in \mathbb{B}_k,\ h \in \mathbb{B}_r.$$

Thus $\deg U < 2^k$ *exactly means that, for each fixed low-bit index x, the coefficient is independent of the high-bit index h.*

Proof. In characteristic two, the image of both 1 and -1 in $\mathbb{F}$ is 1, so (21) loses its signs. □

The theorem includes the endpoint cases. For $k = m$, the high fiber is the singleton $\mathbb{B}_0$ and the condition is vacuous, as every polynomial under consideration has degree below N. For $k = 0$, the theorem says that a polynomial is constant exactly when all full-kernel coefficients form the single signed pattern $(-1)^{m-\mathrm{wt}(h)}\mu_\varnothing$.

4.3 A local projector in kernel coordinates

The same condition can be encoded as a coordinate projector. Recall from Section 3 that for one Boolean coordinate the kernel-to-product coordinate matrix is

$$Z = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}.$$

In product coordinates, retaining only the component whose bit is zero is represented by

$$D = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}.$$

Applying the kernel/product change of coordinates to this elementary degree filter gives

$$Q = Z^{-1} D Z = \begin{pmatrix} 0 & -1 \\ 0 & 1 \end{pmatrix}. \tag{23}$$

The matrix Q is idempotent and its image is the one-dimensional space $\operatorname{span}\{(-1,1)^{\mathsf{T}}\}$.

Proposition 4.4 (Kernel-coordinate projector onto degree below 2^k)**.** *Let Π_k act on the full kernel coefficient array by applying the identity on each of the first k Boolean coordinates and the local map Q of (23) on each of the last $r = m - k$ coordinates. Then $\Pi_k^2 = \Pi_k$, its rank is 2^k, and*

$$\operatorname{im} \Pi_k = \left\{ \lambda : \lambda_{(x,h)} = (-1)^{r-\mathrm{wt}(h)} \mu_x \text{ for some } (\mu_x)_x \right\}.$$

More explicitly,

$$(\Pi_k \lambda)_{(x,h)} = (-1)^{r-\mathrm{wt}(h)} \lambda_{(x,\mathbf{1}_r)}, \tag{24}$$

where $\mathbf{1}_r = (1, \dots, 1)$. Under the kernel-basis isomorphism, the image of Π_k is precisely $\mathbb{F}[X]_{<2^k}$.

Proof. For one high coordinate, $Q(\alpha, \beta)^{\mathsf{T}} = (-\beta, \beta)^{\mathsf{T}}$, so $Q^2 = Q$ and its image is $\operatorname{span}\{(-1,1)^{\mathsf{T}}\}$. Applying this independently on the r high coordinates gives (24), idempotence, and rank 2^k. Its image is therefore exactly the coefficient subspace described in Theorem 4.2, which corresponds to $\mathbb{F}[X]_{<2^k}$. □

In characteristic two,

$$Q = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix},$$

so each high-coordinate projector simply copies the coefficient at bit one to both positions. This is the linear-algebra form of Corollary 4.3.

4.4 What the theorem gives

The Boolean zeta transform converts between the full kernel coefficients and the degree-ordered product coefficients. The theorem above pulls the ordinary condition $\deg U < 2^k$ back through that transform: for each low-bit index, the high-bit coefficients follow an explicit alternating-sign rule. The same rule is represented by a tensor product of local one-dimensional projectors, and the resulting linear combination of full kernels collapses exactly to the truncated kernel family. The next section uses this structure to derive the recursive evaluation algorithm.

5 Recursive evaluation on a quadratic tower

The kernel framework becomes algorithmic when the degree-doubling coordinates come from iterating a single quadratic map. This section derives the exact recursion and the resulting $O(N \log N)$ domain-tower transform.

Let $\mathbb{F}$ be a field and fix $\delta \in \mathbb{F}$. Put

$$\phi(X) = X^2 - \delta, \qquad P_0(X) = X, \qquad P_{i+1}(X) = P_i(\phi(X)). \tag{25}$$

Equivalently, $P_i = \phi^{\circ i}$ for $i \geq 1$, with the convention $P_0(X) = X$. Every P_i is monic of degree 2^i, so the hypotheses of Section 3 apply.

For $m \geq 1$, write a Boolean index as

$$y = (b, z), \qquad b \in \{0, 1\}, \qquad z \in \mathbb{B}_{m-1},$$

where b is the least significant bit. Let $K_z^{P,[m-1]}$ denote the kernel associated with the truncated tower $P_0, \ldots, P_{m-2}$.

5.1 The quadratic kernel recursion

Lemma 5.1 (Shift of the tower through ϕ)**.** *For every $i \geq 1$,*

$$P_i(X) = P_{i-1}(\phi(X)). \tag{26}$$

Consequently,

$$K_{(b,z)}^{P}(X) = (X + b)\, K_z^{P,[m-1]}(\phi(X)). \tag{27}$$

Proof. The first identity is exactly the recurrence (25). Substituting it into

the definition of the full kernel gives

$$\begin{aligned} K^P_{(b,z)}(X) &= (P_0(X) + b) \prod_{i=1}^{m-1} \big(P_i(X) + z_{i-1}\big) \\ &= (X + b) \prod_{j=0}^{m-2} \big(P_j(\phi(X)) + z_j\big) \\ &= (X + b)\, K_z^{P,[m-1]}(\phi(X)). \end{aligned}$$

□

Theorem 5.2 (One-step recursive decomposition)**.** *Let*

$$U(X) = \sum_{b\in\{0,1\}} \sum_{z\in\mathbb{B}_{m-1}} \lambda_{(b,z)} K^P_{(b,z)}(X).$$

Define

$$A(T) = \sum_{z\in\mathbb{B}_{m-1}} \lambda_{(0,z)} K_z^{P,[m-1]}(T), \qquad B(T) = \sum_{z\in\mathbb{B}_{m-1}} \lambda_{(1,z)} K_z^{P,[m-1]}(T). \tag{28}$$

Then

$$U(X) = X\, A(\phi(X)) + (X + 1)\, B(\phi(X)). \tag{29}$$

Equivalently, with

$$C = A + B, \qquad D = B, \tag{30}$$

one has

$$U(X) = D(\phi(X)) + X\, C(\phi(X)). \tag{31}$$

Proof. Apply Lemma 5.1 separately to the $b = 0$ and $b = 1$ parts:

$$\begin{aligned} U(X) &= \sum_z \lambda_{(0,z)} X K_z^{P,[m-1]}(\phi(X)) + \sum_z \lambda_{(1,z)} (X + 1) K_z^{P,[m-1]}(\phi(X)) \\ &= X A(\phi(X)) + (X + 1) B(\phi(X)). \end{aligned}$$

The second form follows by collecting $B(\phi(X)) + X(A + B)(\phi(X))$. □

The decomposition reduces an m-bit kernel expansion to two $(m-1)$-bit kernel expansions evaluated at the common image $\phi(X)$. Thus the coefficient split itself is free of polynomial arithmetic: it is only the partition

$$\lambda \longmapsto \big((\lambda_{(0,z)})_z, (\lambda_{(1,z)})_z\big),$$

followed, if desired, by the pointwise change $(A, B) \mapsto (C, D) = (A + B, B)$.

5.2 Evaluation on paired preimages

Because $\phi(X) = X^2 - \delta$ is even,

$$\phi(u) = \phi(-u)$$

for every $u \in \mathbb{F}$. Write $t = \phi(u)$. Evaluating (31) at the paired preimages u and $-u$ gives

$$\begin{pmatrix} U(u) \\ U(-u) \end{pmatrix} = \begin{pmatrix} 1 & u \\ 1 & -u \end{pmatrix} \begin{pmatrix} D(t) \\ C(t) \end{pmatrix}. \tag{32}$$

When $\operatorname{char} \mathbb{F} \neq 2$ and $u \neq 0$, the local matrix is invertible, with determinant $-2u$. Hence

$$D(t) = \frac{U(u) + U(-u)}{2}, \qquad C(t) = \frac{U(u) - U(-u)}{2u}. \tag{33}$$

The forward recursion itself does not require these inverses and therefore remains meaningful in arbitrary characteristic; only the two-point inverse formula needs $2u \neq 0$.

5.3 Recursive evaluation tree

Suppose a finite evaluation set $\Omega_m \subset \mathbb{F}$ is organized into quadratic fibers and satisfies

$$\phi(\Omega_m) = \Omega_{m-1}, \qquad |\Omega_j| = 2^j, \tag{34}$$

with each point of Ω_{j-1} having two designated preimages in Ω_j. Then (31) gives a recursive evaluator: first evaluate C and D on Ω_{m-1}, then reconstruct the values of U on each fiber using (32).

Proposition 5.3 (Cost of the recursive evaluator)**.** *Under (34), the transform can be evaluated with $O(N \log N)$ field operations for $N = 2^m$, assuming the fiber points and the required multipliers are supplied. More precisely, if one recursive level evaluates two half-size transforms and performs $O(N)$ local arithmetic, the operation count satisfies*

$$T(N) = 2T(N/2) + O(N),$$

and hence $T(N) = O(N \log N)$.

Proof. At one level, the two coefficient arrays defining C and D have size $N/2$. Their recursive evaluations cost $2T(N/2)$. Once those values are available on Ω_{m-1}, each of the $N/2$ fibers is reconstructed using a constant number of field operations, for an additional $O(N)$ cost. The stated recurrence and bound follow. □

The $O(N \log N)$ count covers the recursive field operations once the domain points and stage multipliers are available. Domain construction, precomputation, memory movement, and application-level protocol work are separate costs.

5.4 Connection with the standard degree-two recursion

Equation (31) is the familiar even/odd form of a degree-two recursion, expressed in kernel coordinates. On a fiber $\{u, -u\}$, the local reconstruction matrix is exactly the matrix in (32). Thus the coefficient change $(A, B) \leftrightarrow (C, D)$ puts the kernel recursion into the standard degree-two even/odd form. This exact local change is the form specialized to Mersenne-31 and compared with the recursive Circle-FFT stages in the next section.

6 Mersenne-31 and the Circle-FFT rescaling

We now specialize the quadratic tower of Section 5 to the map

$$\phi(s) = s^2 - 2 \tag{35}$$

over the Mersenne prime field

$$\mathbb{F}_p, \qquad p = 2^{31} - 1.$$

The algebraic identities in this section hold over any field of odd characteristic; the Mersenne-31 specialization is the implementation target used later in the paper.

Let

$$\pi(c) = 2c^2 - 1. \tag{36}$$

For the x-coordinate c of a point on the unit circle, $\pi(c)$ is the x-coordinate after doubling. The affine rescaling

$$s = 2c \tag{37}$$

intertwines the two quadratic maps exactly:

Lemma 6.1 (Exact rescaling of the quadratic maps)**.** *For every c,*

$$\phi(2c) = 2\pi(c). \tag{38}$$

Proof. Directly,

$$\phi(2c) = (2c)^2 - 2 = 4c^2 - 2 = 2(2c^2 - 1) = 2\pi(c).$$

□

Let the degree-doubling tower be

$$P_0(s) = s, \qquad P_{i+1}(s) = P_i(\phi(s)).$$

The same rescaling propagates through every level.

Proposition 6.2 (Tower identity under Circle scaling)**.** *For every $i \geq 0$,*

$$P_i(2c) = 2\,\pi^{\circ i}(c), \tag{39}$$

where $\pi^{\circ 0}(c) = c$.

Proof. For $i = 0$, the identity is $P_0(2c) = 2c$. If it holds at level i, then

$$\begin{aligned} P_{i+1}(2c) &= P_i(\phi(2c)) = P_i(2\pi(c)) \\ &= 2\,\pi^{\circ i}(\pi(c)) = 2\,\pi^{\circ(i+1)}(c), \end{aligned}$$

using Lemma 6.1 and the induction hypothesis. □

6.1 Product-basis equivalence

For $a \in \mathbb{B}_m$, recall

$$B_a(s) = \prod_{i=0}^{m-1} P_i(s)^{a_i}.$$

Define the corresponding Circle-coordinate product

$$C_a(c) = \prod_{i=0}^{m-1} \left(\pi^{\circ i}(c)\right)^{a_i}. \tag{40}$$

Then Proposition 6.2 gives the exact diagonal relation

$$B_a(2c) = 2^{\mathrm{wt}(a)} C_a(c). \tag{41}$$

Thus, after the fixed input rescaling $s = 2c$, the degree-doubling product basis of this paper is the one-variable product basis used by the recursive Circle-FFT stages, up to a diagonal coefficient scaling. Since p is odd, every diagonal factor $2^{\mathrm{wt}(a)}$ is invertible in $\mathbb{F}_p$.

To make the matrix relation explicit, let $\mathcal{C}$ be any ordered list of Circle coordinates and let

$$E_C(\mathcal{C})_{c,a} = C_a(c), \qquad E_B(2\mathcal{C})_{c,a} = B_a(2c).$$

If

$$\Delta_m = \operatorname{diag}\Big(2^{\mathrm{wt}(a)}\Big)_{a\in\mathbb{B}_m},$$

then

$$E_B(2\mathcal{C}) = E_C(\mathcal{C})\Delta_m. \tag{42}$$

Section 3 showed that the Boolean-kernel coordinates are related to the product coordinates by the fixed complemented Boolean-zeta matrix Z_m, independently of the coefficients of the polynomials P_i. Consequently, if $E_K(2\mathcal{C})$ denotes the evaluation matrix whose columns are the full kernels K_y^P evaluated at $s = 2c$, then

$$E_K(2\mathcal{C}) = E_C(\mathcal{C})\Delta_m Z_m. \tag{43}$$

Equation (43) separates the three pieces of the transform: the one-variable Circle-stage product-basis evaluation, a diagonal scaling, and the Boolean-zeta change of coordinates.

6.2 Recursive butterfly under Circle scaling

For the specialization $\phi(s) = s^2 - 2$, the recursion from Theorem 5.2 is

$$U(s) = D(\phi(s)) + s\,C(\phi(s)). \tag{44}$$

Substituting $s = 2c$ and using (38) yields

$$U(2c) = D(2\pi(c)) + 2c\,C(2\pi(c)). \tag{45}$$

If

$$\widetilde{C}(t) = 2C(t),$$

then on the two points c and $-c$ in a fiber of π,

$$\begin{pmatrix} U(2c) \\ U(-2c) \end{pmatrix} = \begin{pmatrix} 1 & c \\ 1 & -c \end{pmatrix} \begin{pmatrix} D(2\pi(c)) \\ \widetilde{C}(2\pi(c)) \end{pmatrix}. \tag{46}$$

Hence the kernel recursion and the standard degree-two one-variable recursion used by the recursive Circle-FFT stages have the same local butterfly after the fixed diagonal normalization of the odd component.

6.3 Complete M31 evaluation-domain tower

The multiplicative group $\mathbb{F}_{p^2}^{\times}$ is cyclic of order $p^2 - 1$. The norm map

$$N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(g) = g^{p+1}$$

is surjective onto $\mathbb{F}_p^{\times}$, so its kernel G has order

$$|G| = \frac{p^2 - 1}{p - 1} = p + 1 = 2^{31}.$$

It is therefore cyclic and contains an element ζ of exact order 2^{31}.

For every integer e, define the trace coordinate

$$x_e = \zeta^e + \zeta^{-e}.$$

Because $\zeta \in G$, one has $\zeta^{p+1} = 1$, hence $\zeta^p = \zeta^{-1}$. Therefore

$$x_e^p = \zeta^{ep} + \zeta^{-ep} = \zeta^{-e} + \zeta^e = x_e,$$

so $x_e \in \mathbb{F}_p$.

For $0 \le j \le 29$, define

$$D_j = \{x_{2^j a} : a \in \mathbb{Z},\ a \text{ odd}\} \subseteq \mathbb{F}_p.$$

The periodicity of powers of ζ makes each D_j finite.

For every exponent e,

$$\begin{aligned} \phi(x_e) &= (\zeta^e + \zeta^{-e})^2 - 2 \\ &= \zeta^{2e} + 2 + \zeta^{-2e} - 2 \\ &= \zeta^{2e} + \zeta^{-2e} \\ &= x_{2e}. \end{aligned}$$

Thus exponent doubling in G induces the affine map ϕ.

Trace-coordinate identification. For all integers e, f,

$$\boxed{x_e = x_f \quad \Longleftrightarrow \quad e \equiv f \text{ or } e \equiv -f \pmod{2^{31}}.}$$

The reverse implication is immediate. For the forward implication, put $t = \zeta^e$ and $u = \zeta^f$. The equality $x_e = x_f$ says

$$t + t^{-1} = u + u^{-1}.$$

Both t and u are roots of

$$T^2 - x_e T + 1.$$

The two roots of this polynomial are t and t^{-1}; hence $u = t$ or $u = t^{-1}$. Since ζ has order 2^{31}, these alternatives are respectively

$$f \equiv e \pmod{2^{31}} \qquad \text{or} \qquad f \equiv -e \pmod{2^{31}}.$$

Cardinalities. Fix $0 \le j \le 29$, and set

$$n = 31 - j.$$

The odd residue classes modulo 2^n number 2^{n-1}. By the trace-coordinate identification,

$$x_{2^j a} = x_{2^j b} \quad \Longleftrightarrow \quad a \equiv \pm b \pmod{2^n}.$$

For $n \ge 2$, negation has no fixed point among the odd residue classes: if $a \equiv -a \pmod{2^n}$, then $2^n \mid 2a$, which is impossible because a is odd and $n \ge 2$. Every equivalence class under $a \sim -a$ consequently has exactly two elements. Therefore

$$\boxed{|D_j| = \frac{2^{n-1}}{2} = 2^{n-2} = 2^{29-j}.}$$

Surjectivity and exact two-to-one branching. For $j < 29$, exponent doubling gives

$$\phi(x_{2^j a}) = x_{2^{j+1} a},$$

and an odd class a remains odd after reduction modulo 2^{30-j}. It follows that

$$\phi(D_j) \subseteq D_{j+1}.$$

Conversely, every element of D_{j+1} has the form $x_{2^{j+1} b}$ for an odd b, and

$$\phi(x_{2^j b}) = x_{2^{j+1} b}.$$

Hence

$$\phi(D_j) = D_{j+1}.$$

The cardinality formula now gives

$$|D_j| = 2|D_{j+1}|,$$

so every fiber has average size two. The fibers are in fact explicitly paired. If

$$\alpha = x_{2^j a},$$

then

$$-\alpha = x_{2^j(a+2^{30-j})},$$

because

$$\zeta^{2^j(a+2^{30-j})} = \zeta^{2^j a+2^{30}} = -\zeta^{2^j a},$$

where $\zeta^{2^{30}} = -1$. Moreover,

$$\phi(-\alpha) = \phi(\alpha).$$

For $j < 29$, these two preimages are distinct. Indeed, if $x_{2^j a} = 0$, then

$$\zeta^{2^{j+1}a} = -1 = \zeta^{2^{30}},$$

and hence

$$2^{j+1}a \equiv 2^{30} \pmod{2^{31}}.$$

Since a is odd, this would force $j = 29$, contrary to $j < 29$. Thus $\alpha \neq 0$, and the odd characteristic of M31 implies $\alpha \neq -\alpha$.

Equivalently, reduction of odd exponents modulo 2^{31-j} to odd exponents modulo 2^{30-j} has exactly two classes modulo sign above each target class. Therefore, for every $0 \leq j < 29$,

$$\boxed{\phi : D_j \longrightarrow D_{j+1} \text{ is exactly two-to-one,}}$$

and each fiber is of the form

$$\phi^{-1}(\beta) \cap D_j = \{\alpha, -\alpha\}.$$

The terminal domain and the repeated preimage. At $j = 29$, the defining odd exponents are taken modulo 4. Let

$$q = \zeta^{2^{29}}.$$

Then q has order 4, so

$$q^2 = -1 \qquad \text{and} \qquad q^{-1} = -q.$$

Consequently,

$$x_{2^{29}} = q + q^{-1} = 0.$$

The two odd residue classes modulo 4 differ by sign and therefore give the same trace coordinate. Hence

$$\boxed{D_{29} = \{0\}.}$$

The next forward value is

$$\phi(0) = -2.$$

The inverse equation over this value is

$$\phi(X) = -2 \quad \Longleftrightarrow \quad X^2 - 2 = -2 \quad \Longleftrightarrow \quad X^2 = 0.$$

It has the single root $X = 0$ with multiplicity two. Equivalently,

$$\phi'(X) = 2X$$

vanishes at $X = 0$, so -2 is the critical value reached from the terminal domain. The paired roots $\{\alpha, -\alpha\}$ have coalesced at $\alpha = 0$.

It follows that the sequence

$$D_0 \xrightarrow{2:1} D_1 \xrightarrow{2:1} \cdots \xrightarrow{2:1} D_{29} = \{0\}$$

contains exactly 29 successive two-to-one maps with two distinct preimages. Since

$$|D_0| = 2^{29}$$

and the next inverse equation has the repeated root $X = 0$, this sequence cannot be extended by another layer with two distinct preimages along the same orbit. Thus $D_0, \ldots, D_{29}$ give the complete distinct-preimage tower along this constructed norm-one-subgroup orbit for the M31 map $X \mapsto X^2 - 2$.

Corollary 6.3 (Instantiation of the recursive evaluation domains)**.** *For every $0 \le m \le 29$, define*

$$\Omega_j = D_{29-j}, \qquad 0 \le j \le m.$$

Then

$$|\Omega_j| = 2^j, \qquad \phi(\Omega_j) = \Omega_{j-1} \quad (1 \le j \le m),$$

and every point of Ω_{j-1} has exactly two distinct preimages in Ω_j. Thus these sets satisfy the domain hypothesis (34)*. In particular, a depth-m transform uses the square base domain*

$$\Omega_m = D_{29-m}, \qquad |\Omega_m| = 2^m.$$

Proof. The cardinality formula gives $|D_{29-j}| = 2^j$. For $j \ge 1$, the two-to-one map $\phi : D_{29-j} \to D_{30-j}$ proved above is exactly $\phi : \Omega_j \to \Omega_{j-1}$. □

6.4 Relation to the recursive Circle-FFT stages

The identities above establish the exact relationship needed for the implementation comparison. The rescaling $s = 2c$ sends $\phi(s) = s^2 - 2$ to the Circle x-coordinate map $\pi(c) = 2c^2 - 1$; the product-basis evaluation matrices differ only by a diagonal scaling; and the local recursive butterflies agree after the same normalization. These are precisely the recursive one-variable stages shared with the Circle-FFT computation. The complete Circle FFT additionally contains its circle-to-line step, point ordering, evaluation subsets, stage-multiplier conventions, and implementation layout.

7 Correctness and implementation checks

The implementation evidence in this section is used as a validation of the mathematical transform developed in the preceding sections, not as a substitute for the algebraic proofs. The optimized path used in the reported measurements is `kernel_raw_neon_r8_tail2` in `crates/stwo/examples/kernel_vs_stwo.rs`. The comparison baseline calls Stwo's public SIMD Circle-FFT entry point `rfft::fft`.

7.1 Source-level transform organization

The implementation represents the native recursive coefficients in bit-reversed order. After the one-coordinate linear change

$$(\lambda_0, \lambda_1) \longmapsto (D, C) = (\lambda_1, \lambda_0 + \lambda_1),$$

the local evaluation step is exactly

$$(D, C) \longmapsto (D + \alpha C,\ D - \alpha C), \tag{47}$$

and therefore uses one field multiplication and two additions/subtractions per butterfly once the multiplier α is available. The source performs the kernel-to-native coefficient conversion as correctness/preparation work outside the timed native-basis transform.

The final AArch64 path uses raw M31 words and consumes a precomputed array of level multipliers. Its schedule has three source-visible components.

1. While at least five levels remain, the routine applies `neon_radix8_block`. One call keeps the data for three dependent binary levels inside a radix-8 block and consumes the parent, two child, and four grandchild multipliers associated with those levels.

2. Any remaining levels above the final two are executed by the single-level NEON routine `neon_level4`.

3. When exactly two levels remain, the implementation calls `neon_tail2_fused`, which replaces the final scalar tail by a fused two-level NEON kernel. The generic scalar fallback in the same function is not taken for the benchmarked sizes $\log_2 N \in \{18, 19, 20\}$.

This source-level description is paired with an end-to-end exact-output check of the complete optimized path. That check jointly exercises the radix-8 fusion, raw-word arithmetic, multiplier indexing, and NEON vector mapping used by the tested implementation.

7.2 Exact-output correctness check

Before interpreting timing data, the actual `kernel_raw_neon_r8_tail2` path was checked against a scalar reference implementing the affine recursion used by the M31 specialization. The gate invokes the production benchmark path itself rather than a reimplementation of its schedule. It therefore exercises, jointly, the fused level schedule, the consumed level multipliers, the raw-M31 arithmetic, and the NEON tail reached by that function on the tested sizes.

The exact-output correctness check covered 543 vectors: all tested vectors for

$$\log_2 N = 2, \ldots, 8,$$

together with deterministic vectors at

$$\log_2 N \in \{9, 10, 12, 14, 16, 18, 20\}.$$

Equality was checked as exact M31 field values. On this verification set, the optimized path agrees with the scalar reference for the quadratic recursion.

This exact-output check is executable evidence for the stated verification set. The algebraic correctness of the recursive transform is established independently in the preceding sections.

7.3 Measurement and reproducibility boundary

The benchmark data reported below were collected under the stated configuration. The timed regions are deliberately narrow. Domain construction, multiplier generation or precomputation, benchmark input preparation,

benchmark-level allocation and setup, warmup, and output interpretation are outside the timed regions.

On the kernel side, `kernel_raw_neon_r8_tail2` first copies the prepared input into the output buffer and then runs the in-place transform; this copy is therefore inside the measured kernel call. On the Stwo side, the measured call is the public SIMD `rfft::fft` entry point. Consequently, the reported numbers are transform-kernel measurements: they are not end-to-end prover timings and they do not measure basis construction plus transform plus output conversion.

The benchmark helper performs two warmup calls before collecting the timed repetitions for a round and reports the median elapsed time of those timed calls. The stability experiment uses seven independent rounds with nine timed repetitions per round at each reported size.

The two timed regions are not asserted to have identical memory traffic. The purpose of the comparison is narrower: to determine whether the mathematically derived kernel transform reaches the same native performance regime as the pinned Stwo SIMD Circle-FFT implementation under the stated boundary.

The comparison uses the Stwo source revision recorded in the reproducibility metadata. The benchmark environment records Apple M1 / ARM64 hardware, Rust `1.96.0-nightly`, Cargo `1.96.0-nightly`, source-state and file-hash provenance. These identifiers are reproducibility metadata rather than mathematical assumptions.

8 Benchmark results

We compare the optimized kernel path `kernel_raw_neon_r8_tail2` against the SIMD implementation from `starkware-libs/stwo` at the source revision recorded in the reproducibility metadata. The tested sizes were exactly

$$(\log_2 N, N) \in \{(18, 262144), (19, 524288), (20, 1048576)\}.$$

For each size, the stability experiment contains seven independent rounds with nine repetitions per round.

At $\log_2 N = 20$, the aggregate kernel and Stwo medians were respectively

$$6.924750\,\text{ms} \qquad \text{and} \qquad 7.058000\,\text{ms}.$$

The median of the per-round kernel/Stwo ratios was

$$0.979510,$$

$\log_2 N$	N	Kernel (ms)	Stwo (ms)	Ratio
18	262144	1.536916	1.542792	0.996191
19	524288	3.218542	3.307583	0.974495
20	1048576	6.924750	7.058000	0.979510

Table 1: Timing summary across seven independent rounds. Each displayed kernel or Stwo median is the median of the seven per-round medians. The ratio is the median of the seven per-round kernel/Stwo ratios.

with a per-round range

$$[0.976630, 0.996149].$$

All seven rounds at this size were within five percent.

The ratio column in Table 1 is the median of the per-round ratios; it is not, in general, the quotient of the two displayed aggregate medians. This distinction explains why 6.924750/7.058000 is not exactly 0.979510.

We therefore summarize the result as stable, similar transform-kernel running times on the recorded Apple ARM64 configuration. The approximately two-percent difference between the displayed aggregate medians lies inside the observed round-to-round range reported above.

Implementation variants. The measurements compare three kernel variants: `scalar_r8`, `old_raw_neon`, and `tail2_neon`. At $\log_2 N = 20$, their across-round median times were respectively

$$9.500417\,\text{ms}, \qquad 9.144209\,\text{ms}, \qquad 6.924750\,\text{ms}.$$

Among these variants, the largest measured change is associated with the transition from `old_raw_neon` to `tail2_neon`, rather than merely from the scalar path to the earlier NEON path.

9 Discussion and limitations

The algebraic and systems results answer different parts of the same question. The Boolean-zeta/kernel construction describes ordinary polynomial degree in a recursive coefficient system over arbitrary fields. The quadratic recursion adds an $O(N \log N)$ domain-tower transform. The M31 specialization supplies a concrete 29-level evaluation tower and an exact rescaling to the one-variable recursion used by Circle FFTs. The ARM64 experiments then test one software realization of that structure.

The performance measurements are intentionally defined at the transform-kernel boundary. Domain and stage-multiplier construction, benchmark preparation, warmup, setup, and output interpretation are outside the timed regions. The kernel timing includes its input-to-output copy, while the Stwo measurement is the public SIMD `rfft::fft` call. The comparison should therefore be read as a comparison of these two native transform calls under the stated boundary.

On the recorded Apple M1 / ARM64 system, all seven rounds at $\log_2 N \in \{18, 19, 20\}$ remain within five percent of the Stwo baseline. At $N = 2^{20}$ the median per-round kernel/Stwo ratio is 0.979510, with observed range $[0.976630, 0.996149]$. These numbers are specific to this machine, toolchain, implementation revision, and transform boundary. Measurements on other processors, end-to-end proof systems, and future FPGA implementations are separate experiments.

The main practical lesson is that the Boolean-zeta/kernel representation can be carried through the quadratic M31 recursion without losing the performance scale of the selected production Circle-FFT baseline. The next systems test is to implement the complete domain-tower transform on FPGA, including ordering, stage multipliers, and input/output interfaces, and compare it bit-for-bit with the Rust reference before reporting hardware performance.

10 Future work

The present paper stops at a complete software realization and a narrowly defined native-transform comparison. Two follow-on directions are especially natural.

Complete FPGA realization of the domain-tower FFT. The first systems objective is a bit-exact FPGA implementation of the complete M31 transform studied here, rather than an isolated arithmetic microbenchmark. A reproducible hardware realization should include the construction or loading of the evaluation-domain and stage-multiplier data, the required input coordinate preparation, every level of the recursive transform, and the exact input/output ordering and permutations. Correctness should be checked bit-for-bit against the Rust reference on deterministic and randomized test vectors before any performance result is accepted. Hardware claims should then be reported only from same-device implementation evidence, including at least lookup-table (LUT), flip-flop (FF), DSP-block and block-

RAM (BRAM) use, achieved clock frequency, latency, throughput, and, when available, power. This would determine whether the regular degree-two dataflow and the local one-multiplication-plus-two-addition butterfly retain their software-level advantages under an explicitly pipelined hardware schedule.

Connection to multilinear Sumcheck machinery. A second direction is to make precise the relationship between the Boolean-zeta/kernel coordinates developed here and the recursive tree representation of multilinear polynomials used for Sumcheck. The earlier Sumcheck work explicitly leaves as future work the design of transformations from its recursive multilinear representation to univariate representations compatible with folding-based protocols such as the Fast Reed–Solomon Interactive Oracle Proof of Proximity (FRI) [13]. The next step should therefore be algebraic, not merely architectural: define an explicit map from the multilinear tree coefficients to the present kernel/zeta coordinates, prove which degree, evaluation, and folding invariants it preserves, and identify the precise boundary at which a univariate low-degree test becomes applicable. Only after those invariants are established does it become meaningful to investigate a shared M31 FPGA datapath for Sumcheck reductions and the domain-tower transform. Constructing and validating that Sumcheck–FRI bridge and the corresponding shared hardware pipeline is part of this future program.

11 Conclusion

This paper connects the Boolean zeta transform to a family of kernel polynomials built from monic coordinates of degrees $1, 2, 4, \ldots, 2^{m-1}$. If

$$U = \sum_y \lambda_y K_y^P,$$

the kernel coefficients are converted to the degree-ordered product coefficients by a fixed complemented Boolean zeta transform. Pulling the ordinary condition $\deg U < 2^k$ back through that transform gives an explicit alternating-sign rule on the high-bit coefficients. The same condition is represented by a tensor product of local one-dimensional projectors, and in characteristic two it becomes simple constancy as the high bits vary.

For coordinates generated by a quadratic map, the kernel expansion has the recursive form

$$U(X) = D(\phi(X)) + X\, C(\phi(X)),$$

so evaluation on each pair of preimages is a degree-two butterfly. When the evaluation sets form a two-to-one domain tower this gives an $O(N \log N)$ transform. Over M31 with $\phi(s) = s^2 - 2$, the rescaling $s = 2c$ turns the map exactly into $c \mapsto 2c^2 - 1$, the one-variable recursion used by the recursive stages of Circle FFTs. The norm-one-subgroup construction supplies 29 levels with two distinct preimages at every level.

The optimized Rust/ARM64 implementation agrees exactly with the scalar quadratic-recursion reference on the recorded 543-vector test set. On the tested Apple M1 system, the seven-round measurements at $\log_2 N \in \{18, 19, 20\}$ place its transform-kernel running time in the same range as the pinned Stwo SIMD Circle-FFT call. At $N = 2^{20}$ the median per-round kernel/Stwo ratio is 0.979510.

The resulting chain is explicit: the Boolean zeta transform gives the kernel coefficient map, the degree theorem identifies exactly which coefficient arrays represent degree below 2^k, the quadratic recursion turns those coordinates into a domain-tower FFT, and the M31 rescaling connects that recursion to the one-variable stages used in Circle FFT implementations.

Statements and declarations

Competing interests. The author has no competing interests to declare.

Funding. No external funding was received for conducting this study.

Code and data availability. The implementation source, exact-output checks, experiment summaries, benchmark evidence, paper sources, and reproducibility metadata are available in the public Algorizk Labs repository `https://github.com/qoosmo/zeta-kernel-m31-fft`. The repository records the upstream Stwo source revision used by the benchmark, while the manuscript states performance claims only at the experimental boundary defined in Sections 7 and 8.

Author contributions. Ali Mkhida: conceptualization, methodology, mathematics, software, experimentation, validation, writing, and supervision.

Generative AI. The author used DeepSeek, GPT, and Claude for implementation assistance, LaTeX formatting, and technical writing support. All generated content was reviewed and the author takes full responsibility.

A Notation and implementation terms

This appendix collects notation and implementation terminology used across the paper. It is intended as a navigation aid; the mathematical definitions and proofs remain in the main text.

Boolean indices. For $m \geq 0$, the Boolean cube is $\{0,1\}^m$. If $a = (a_0, \ldots, a_{m-1})$, then

$$|a|_2 = \sum_{i=0}^{m-1} a_i 2^i$$

denotes its binary value and $\bar{a} = (1-a_0, \ldots, 1-a_{m-1})$ its Boolean complement. The Hamming weight is $\mathrm{wt}(a) = \sum_i a_i$. The symbol $\mathbf{1}[\mathcal{P}]$ is the indicator of a proposition $\mathcal{P}$.

Polynomial spaces and composition. For a field $\mathbb{F}$, $\mathbb{F}[X]_{<d}$ denotes the vector space of polynomials of degree strictly less than d, with the convention $\deg 0 = -\infty$. For a polynomial map ϕ, the notation $\phi^{\circ i}$ denotes its i-fold composition, not an ordinary power. The notation $[X^j]F$ denotes the coefficient of X^j in F, and $\mathrm{lc}(F)$ denotes its leading coefficient. A unitriangular matrix is a triangular matrix whose diagonal entries are all one.

Tensor notation. The symbol $\otimes$ denotes the Kronecker (tensor) product of matrices or linear maps, and I_2 denotes the 2×2 identity matrix. Tensor factorizations in the paper always use the Boolean-coordinate ordering fixed in the kernel-framework section.

M31 and quadratic extensions. M31 is the prime field $\mathbb{F}_p$ with $p = 2^{31} - 1$. The group $\mathbb{F}_{p^2}^\times$ is the multiplicative group of the quadratic extension. For $z \in \mathbb{F}_{p^2}$,

$$\mathrm{Tr}(z) = z + z^p, \qquad \mathrm{N}(z) = z^{p+1}$$

are the field trace and norm. Standard finite-field facts used for these maps are recalled from [10].

Circle notation. When an ordered set of Circle x-coordinates is denoted by $\mathcal{C} = (c_j)_j$, the notation $2\mathcal{C}$ means the ordered scaled list $(2c_j)_j$. It does not denote a sumset. The M31 comparison in the paper concerns the one-variable recursion obtained after the circle-to-line projection. A complete Circle FFT contains additional stages and ordering conventions.

Paired-preimage domain tower. A domain tower $\Omega_m \to \Omega_{m-1} \to \cdots \to \Omega_0$ is called paired-preimage when the quadratic map ϕ maps each Ω_j onto Ω_{j-1} and every point at level $j-1$ has exactly two distinct preimages at level j. Evaluation on such a pair produces the local degree-two *butterfly* used by the recursive transform.

Stage multipliers and twiddles. At a paired-preimage butterfly, the field element multiplying the odd/native coordinate is called the stage multiplier. In FFT terminology it plays the role of a twiddle factor. The implementation precomputes these values outside the timed transform region.

Bit-reversed order. Bit-reversed order stores an index by reversing the relevant binary digits of its natural-order index. The optimized implementation uses this ordering to make recursively related coefficient blocks and their stage multipliers available in a regular traversal.

ARM64, AArch64, SIMD, and NEON. ARM64 and AArch64 refer to the 64-bit ARM architecture. SIMD means single-instruction, multiple-data execution. NEON is ARM's SIMD instruction set used by the optimized path. A radix-8 block in this paper is an implementation schedule that fuses three dependent binary transform levels; the underlying mathematical transform remains the same binary recursion.

References

[1] Eli Ben-Sasson, Dan Carmon, Swastik Kopparty, and David Levit. Elliptic curve fast fourier transform (ecfft) part i: Fast polynomial algorithms over all finite fields, 2021.

[2] Eli Ben-Sasson, Dan Carmon, Swastik Kopparty, and David Levit. Scalable and transparent proofs over all large fields, via elliptic curves: (ecfft part ii). In *Theory of Cryptography*, pages 467–496, 2022.

[3] Dan Carmon, Lior Goldberg, Ulrich Haböck, Leonardo Lerer, Ilya Lesokhin, Shahar Papini, and Shahar Samocha. S-two whitepaper. Cryptology ePrint Archive, Paper 2026/532, 2026.

[4] Nicholas Coxon. Fast transforms over finite fields of characteristic two. *Journal of Symbolic Computation*, 104:824–854, 2021.

[5] Shuhong Gao and Todd D. Mateer. Additive fast fourier transforms over finite fields. *IEEE Transactions on Information Theory*, 56(12):6265–6272, 2010.

[6] Ulrich Haböck. A note on the g-fft. Cryptology ePrint Archive, Paper 2024/1036, 2024.

[7] Ulrich Haböck, David Levit, and Shahar Papini. Circle starks. Cryptology ePrint Archive, Paper 2024/278, 2024.

[8] Ulrich Haböck, Daniel Lubarov, and Jacqueline Nabaglo. Reed-solomon codes over the circle group. Cryptology ePrint Archive, Paper 2023/824, 2023.

[9] Songsong Li and Chaoping Xing. Fast fourier transform via automorphism groups of rational function fields. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3836–3859. SIAM, 2024.

[10] Rudolf Lidl and Harald Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2 edition, 1997.

[11] Sian-Jheng Lin, Tareq Y. Al-Naffouri, Yunghsiang S. Han, and Wei-Ho Chung. Novel polynomial basis with fast fourier transform and its application to reed–solomon erasure codes. *IEEE Transactions on Information Theory*, 62(11):6284–6299, 2016.

[12] Sian-Jheng Lin, Wei-Ho Chung, and Yunghsiang S. Han. Novel polynomial basis and its application to reed–solomon erasure codes. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 316–325. IEEE, 2014.

[13] Ali Mkhida and Adil Iguider. Multilinear polynomials via tree-based circuit and the sumcheck protocol. IACR Cryptology ePrint Archive, Report 2026/1469, 2026.

[14] Gian-Carlo Rota. On the foundations of combinatorial theory i. theory of möbius functions. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 2(4):340–368, 1964.

[15] StarkWare. Stwo: Starkware's next-generation prover. Software repository; accessed 2026-09-01.