Bilinear Hulls for Support Splitting: Characterization, Optimization, and Applications

Yusuke Aikawa[1], Keita Ishizuka[2], and Tomoki Moriya[2]

[1] The University of Tokyo, Bunkyo, Tokyo, Japan
aikawa@mist.i.u-tokyo.ac.jp
[2] Mitsubishi Electric Corporation, Kamakura, Kanagawa, Japan
keitaishizuka1994@gmail.com,
Moriya.Tomoki@bp.MitsubishiElectric.co.jp

Abstract. Sendrier's Support Splitting Algorithm (SSA) is the standard solver for the Permutation Equivalence Problem (PEP); its cost is dominated by a term that grows exponentially in the hull dimension. We study, mathematically, how much this cost can be lowered by replacing the standard inner product, on which the hull is defined, with an alternative bilinear form. Two structural results give a complete answer: only a narrow two-parameter family of bilinear forms produces a hull on which SSA can still operate, namely those whose matrix in the standard basis is a linear combination of the identity and the all-ones matrix; and even within this family the hull dimension drops by at most one (with the reduction achieved on every self-dual code whose length is coprime to the field characteristic). Together, these results give a tight upper bound: since SSA on a code with hull dimension h has cost dominated by q^h, replacing the standard hull with Hull_M lowers this dominant term from q^h to at best q^{h-1}, so no bilinear-form generalization of the inner product can reduce the SSA cost by more than a factor of q, the field size. Going beyond this requires invariants outside the bilinear framework. As an application, we verify that PEP-based SPECK admits the factor-q cost reduction, while LEP-based LESS and PKP-based PERK lie outside our framework. A SageMath implementation of the bilinear-hull preprocessing confirms feasibility at SPECK's recommended length $n = 252$.

Keywords: Code-based cryptography · SPECK signature · Support splitting algorithm · Permutation equivalence · Hull

1 Introduction

Post-quantum digital signatures are an active research area within the ongoing NIST standardization effort, and a number of code-based signature schemes have been proposed in recent years that derive security from hard algorithmic problems on linear codes, most notably from the family of *code equivalence* (and related kernel) problems. Three of the most actively developed proposals sit on

distinct points of this spectrum: LESS [8] relies on the Linear Equivalence Problem (LEP), PERK [1] on the Permuted Kernel Problem (PKP), and SPECK [3], proposed in 2025 by Baldi et al., on a newly introduced variant called Permutation Equivalence of Codes and Kernel (PECK), which in the High q regime is computationally equivalent to the classical Permutation Equivalence Problem (PEP).

PEP and SSA: thirty years of study. PEP and its dedicated solver, Sendrier's *Support Splitting Algorithm* (SSA) [19], have been studied for almost three decades. Petrank and Roth [16] showed that Graph Isomorphism (GI) reduces to PEP, giving the first complexity-theoretic lower bound. For codes with trivial hull, Bardet, Otmani, and Saeed-Taha [5] proved the converse: PEP on linear-complementary-dual (LCD) instances reduces back to GI via orthogonal projectors. Combined with Babai's quasipolynomial-time algorithm for GI [2], this means that PEP on trivial-hull instances is no harder than GI, so trivial-hull codes are an unsuitable parameter choice for conservative PEP-based cryptography. PEP-based schemes must consequently use codes with *large* hull dimension; SPECK pushes this to the extreme by using self-dual codes (hull dimension $h = n/2$) in its proposed parameters.

The cost of SSA is governed by the hull dimension: its time complexity is $O(n^3 + n^2 q^h \log n)$, where $h = \dim \mathrm{Hull}(C)$ and $\mathrm{Hull}(C) = C \cap C^{\perp}$ [19]. Sendrier [18] characterized the average behavior of $\dim \mathrm{Hull}$ for random linear codes, identifying it as the natural invariant governing the cost of SSA; analogous statistics have been derived for cyclic codes [21] and for Hermitian hulls of constacyclic codes [11]. SSA has also been deployed in concrete cryptanalysis: Loidreau and Sendrier [13] used it to detect weak McEliece keys, and Panny [14] recently revisited McEliece secret-key search through the structure of equivalent keys. Across all this work, $\dim \mathrm{Hull}$ has been one of the central parameters whose reduction translates directly into a lower SSA cost.

For the parameter ranges relevant to cryptography, the q^h term in SSA's complexity dominates; any reduction of h therefore directly lowers the cost of the SSA attack. The natural question is whether the hull, defined via the *standard* inner product $\langle x, y \rangle = \sum_i x_i y_i$, is the only permutation-invariant hull arising from a bilinear form.

The bilinear-form route. Identifying a bilinear form with its matrix in the standard basis, we write its *structure matrix* as M and obtain a hull-like object $\mathrm{Hull}_M(C) = C \cap C^{\perp_M}$ for any nondegenerate M. If some choice of M were to produce a smaller hull while remaining compatible with SSA, the q^h cost of the standard PEP solver would drop accordingly. We are led to two natural questions:

(a) *Is there any non-trivial M for which* Hull_M *remains a usable SSA invariant?*
(b) *If so, by how much can the hull dimension be reduced?*

This paper answers both questions and explores the consequences for current code-based signature candidates.

Our contributions.

1. *Structural theory of bilinear hulls (Section 3).* Two structural results delimit the bilinear-form approach: (i) Hull_M is invariant under coordinate permutations (hence SSA-compatible) if and only if $M = aI + bJ$ for some $a, b \in \mathbb{F}_q$, where I is the identity matrix and $J = \mathbf{1}\mathbf{1}^\top$ is the all-ones matrix (every entry equal to 1); (ii) for any such M, $\dim \mathrm{Hull}_M(C)$ differs from $\dim \mathrm{Hull}(C)$ by *at most one*, with the reduction achieved on every self-dual code whose length is coprime to the field characteristic. The proofs use only elementary linear algebra (and a short character argument on S_n).
2. *A tight upper bound on the bilinear-form approach.* Combining (i) and (ii) yields the central conclusion of the paper: *no bilinear-form generalization of the inner product can lower the SSA cost by more than a factor of q.* Going beyond this requires invariants outside the bilinear framework, such as higher-order tensor invariants or non-bilinear algebraic structures. We view this as a clean mathematical limit, not as a security statement about any particular scheme.
3. *Applications to current code-based signature candidates (Sections 4 and 5).* For PEP-based SPECK [3] (via its PECK-to-PEP reduction in the High q regime), the framework yields an SSA attack with cost $O(n^3+n^2q^{n/2-1}\log n)$, a factor of q below standard SSA; at recommended $q = 8861$ this is a factor $q \approx 2^{13}$ in the SSA enumeration term. We emphasize that this is a reduction in SSA *cost* on the underlying PEP instance, not an improvement on the best known attack against SPECK as a scheme. For LEP-based LESS, we prove (Proposition 3) that $M = aI + bJ$ loses its invariance under monomial transformations, so the technique does not extend. For PKP-based PERK, hulls play no role and the framework does not apply at all.
4. *Implementation and empirical validation (Section 6).* We implement the bilinear-hull preprocessing in SageMath and run it on 66 self-dual codes, including SPECK instances ($n = 252$, $q \in \{127, 8861\}$), to validate the predicted dimension reduction and measure the preprocessing overhead. The fixed choice $M = I + J$ achieves the predicted dimension reduction in every trial and completes in 2–5 ms, confirming that the factor-q SSA cost reduction carries no hidden preprocessing overhead.

The contributions above are at the structural level: a complete characterization of the bilinear-form route, a tight quantitative bound on the achievable SSA cost reduction, an applicability survey across current code-based signature candidates, and an implementation confirming that the bilinear-hull preprocessing is practically negligible.

Related work. The standard hull and its statistical properties have been studied extensively for random codes [18], cyclic codes [21], and constacyclic codes with Hermitian duality [11]. Our results are the first to characterize *which* alternative bilinear forms produce permutation-invariant hulls, and to bound the resulting dimension change. Closest in spirit is the recent hull-variation problem of Ho and Johnsen [10] for vector rank-metric codes, which asks how $\dim \mathrm{Hull}$ varies

across an equivalence class of codes. Our perspective is complementary: we fix the code and vary the bilinear form, optimizing the invariant that governs the cost of SSA. Hull-type invariants for matrix codes have also recently been used by Couvreur and Levrat [9] to attack the Matrix Code Equivalence Problem underlying MEDS and ALTEQ.

Concurrent work by Baldi et al. [4] exploits the self-orthogonal structure of public keys in SPECK, earlier versions of LESS, and ABL for compression rather than cryptanalysis. The interaction between such compressed representations and our bilinear hull is left open. On the cost-reduction side, our factor-q result fits the line of concrete refinements such as Panny's recent work on McEliece [14], where polynomial or constant-factor improvements without changing the asymptotic exponent are practically meaningful.

Preliminary version. A subset of the structural results in Section 3 appeared in a short poster abstract. The citation is omitted to preserve double-blind review and will be restored in the camera-ready version.

Organization. Section 2 recalls the necessary background on linear codes, PEP, and SSA. Section 3 develops the structural theory of permutation-invariant bilinear hulls, with full proofs of the characterization, the dimension bound, and the self-dual reduction. Section 4 presents the SPECK signature scheme, the PECK-to-PEP reduction in the High q regime, the bilinear-hull SSA, its cost analysis, and the concrete numerical impact on SPECK's recommended parameters. Section 5 analyzes the technique's applicability to LESS and PERK. Section 6 reports experimental validation at SPECK parameters. Section 7 concludes.

2 Preliminaries

2.1 Linear Codes and Hulls

Let $\mathbb{F}_q$ denote the finite field of order $q = p^m$, where p is prime. An $[n, k]_q$ *linear code* C is a k-dimensional subspace of $\mathbb{F}_q^n$. A *generator matrix* $G \in \mathbb{F}_q^{k\times n}$ is a matrix whose rows form a basis of C. A *parity-check matrix* $H \in \mathbb{F}_q^{(n-k)\times n}$ satisfies $GH^\top = \mathbf{0}$.

The *dual code* of C is $C^\perp = \{y \in \mathbb{F}_q^n : \langle x, y\rangle = 0 \text{ for all } x \in C\}$, where $\langle x, y\rangle = \sum_{i=1}^n x_i y_i$ is the standard inner product. The *hull* of C is defined as

$$\mathrm{Hull}(C) = C \cap C^\perp.$$

A code is *self-orthogonal* if $C \subseteq C^\perp$, and *self-dual* if $C = C^\perp$ (which forces $k = n/2$). For a self-dual code, $\mathrm{Hull}(C) = C$ and hence $\dim \mathrm{Hull}(C) = k = n/2$. If $\mathrm{Hull}(C) = \{\mathbf{0}\}$, the code C is said to have *trivial hull*, and is called *linear complementary dual* (LCD),

Code equivalence. Two notions of code equivalence are of interest. Two codes $C, C' \subseteq \mathbb{F}_q^n$ are *permutation-equivalent* if $C' = \pi(C)$ for some $\pi \in S_n$, and *monomially equivalent* (or *linearly equivalent*) if $C' = \mu(C)$ for some monomial transformation $\mu = D \cdot \pi$, where $D = \mathrm{diag}(d_1, \ldots, d_n)$ with $d_i \in \mathbb{F}_q^*$ is a diagonal scaling. Permutation equivalence is the more restrictive notion, and the hull is invariant under permutations but not under arbitrary monomial transformations. This distinction will matter in Section 5 when we contrast PEP-based and *LEP*-based schemes.

The hull dimension controls the complexity of solving the Permutation Equivalence Problem, as we review next.

2.2 The Permutation Equivalence Problem

Let S_n denote the symmetric group on $\{1, \ldots, n\}$. A permutation $\pi \in S_n$ acts on row vectors of $\mathbb{F}_q^n$ by right multiplication with the corresponding permutation matrix $P = P_\pi \in \mathbb{F}_q^{n \times n}$, $(P)_{ij} = \delta_{j,\pi(i)}$, so that $\pi(x) = xP$ and $(\pi(x))_j = x_{\pi^{-1}(j)}$. The action on a code is $\pi(C) = CP = \{xP : x \in C\}$. We will use freely $PP^\top = I$ (orthogonality) and $P\mathbf{1} = \mathbf{1}$, $\mathbf{1}^\top P = \mathbf{1}^\top$ (the all-ones vector $\mathbf{1} = (1, \ldots, 1)$ is fixed).

Definition 1 (Permutation Equivalence Problem (PEP)). *Given generator matrices G, G' of two permutation-equivalent $[n, k]_q$ codes C, C', find $\pi \in S_n$ such that $C' = \pi(C)$.*

PEP and Graph Isomorphism. PEP has been studied for almost three decades. Petrank and Roth [16] showed that PEP is at least as hard as Graph Isomorphism (GI), giving the first complexity-theoretic lower bound on the problem. For codes with trivial hull (LCD codes), Bardet, Otmani, and Saeed-Taha [5] proved the converse direction: PEP on LCD instances reduces to GI via orthogonal projectors. Combined with Babai's quasipolynomial-time algorithm for GI [2], this means PEP on small-hull instances admits a quasipolynomial-time solver and is therefore unsuitable as the basis of a code-based signature scheme. PEP-based schemes must consequently use codes with *large* hull dimension; SPECK pushes this to the extreme by using self-dual codes (hull dimension $n/2$). A comprehensive survey of the hardness landscape for code equivalence problems can be found in [6].

The hull as the SSA invariant. A key property of the hull is its invariance under permutations: $\mathrm{Hull}(\pi(C)) = \pi(\mathrm{Hull}(C))$ for all $\pi \in S_n$, so any π sending C to C' also sends $\mathrm{Hull}(C)$ to $\mathrm{Hull}(C')$. The statistical behavior of $\dim \mathrm{Hull}$ for random linear codes was characterized by Sendrier [18] and extended to structured families such as cyclic codes by Skersys [21]; alternative dualities (Hermitian) were studied by Jitman and Sangwisut [11]. These results identify the standard hull as the natural cryptographic invariant for PEP: its dimension governs the cost of the only known PEP-specific solver, the Support Splitting Algorithm.

2.3 The Support Splitting Algorithm

Sendrier's *Support Splitting Algorithm* (SSA) [19] solves PEP by computing *signatures*: invariant maps S that assign to each coordinate of a code an element of some set E such that

$$S(\pi(C), \pi(i)) = S(C, i) \qquad \text{for all codes } C,\ i \in [n],\ \pi \in S_n. \tag{1}$$

By matching signatures of coordinates between C and C', SSA recovers π. The construction builds on Leon's earlier work on code automorphism groups [12].

Why the hull dimension governs the cost. The signature used in practice combines puncturing with weight enumeration: for each coordinate i, one punctures the code at i to obtain $C \setminus \{i\}$, and records its weight enumerator (or a hash thereof) as the signature of position i. Computing the weight enumerator of an $[n, k']_q$ code requires enumerating up to $q^{k'}$ codewords, so *the dimension of the code on which SSA operates is the dominant cost factor*. Since $\mathrm{Hull}(C)$ is permutation-invariant, SSA can be applied to the hull rather than to the full code. When $\dim \mathrm{Hull}(C) = h < k$, this reduces the search space from q^k to q^h. The complexity of SSA on an $[n, k]_q$ code with hull dimension h is

$$O(n^3 + n^2 q^h \log n), \tag{2}$$

where the q^h term, arising from enumeration in signature computation, dominates for cryptographic parameters [19]. For self-dual codes ($h = k = n/2$), the hull provides no reduction and SSA has complexity $O(n^3 + n^2 q^{n/2} \log n)$.

SSA in cryptanalysis. SSA has been employed in concrete cryptanalysis: Loidreau and Sendrier [13] used it to detect weak keys in the McEliece cryptosystem, and Panny [14] recently revisited McEliece secret-key search via the structure of equivalent keys, lowering the concrete cost even though the asymptotic exponent is unchanged. This line of work illustrates that even constant-factor improvements to the SSA cost can be meaningful in concrete cryptanalysis, which is the frame in which we state our factor-q result.

Remark 1 (Related solvers for code equivalence). SSA is specific to PEP and its cost is governed by the hull dimension. For the strictly more general Linear Equivalence Problem (LEP), the dedicated solvers are based on low-weight codeword search and collision search and do *not* depend on the hull. This line originates with Leon's low-weight-codeword algorithm [12]; Beullens [7] improved it via collision search, and it has since been further improved by the recent algorithms of Bardet, Brion, Otmani, Saeed, and Sendrier [**?**] and of Budroni and Esser [**?**], which are the current state of the art. A reduction from LEP to PEP exists via the closure construction of Sendrier and Simos [20,15], but it inflates the code length from n to $n(q-1)$ and is outperformed by these dedicated LEP solvers. Moreover, for $q \geq 5$ the closure of an LEP instance is always weakly self-dual, so the resulting PEP instance has near-maximal hull dimension and SSA

performs poorly on it regardless of the bilinear form used. This distinction will be important when we discuss the boundary of our technique in Section 5: hull-based techniques target PEP specifically, whereas LEP is solved by hull-agnostic collision algorithms.

3 Permutation-Invariant Bilinear Hulls

This section develops the structural theory underlying the cost reduction of Section 4. We generalize the hull via alternative bilinear forms (Section 3.1), characterize which generalizations remain compatible with SSA (Theorem 1), and bound the resulting hull dimension (Theorem 2 and Corollary 1). The proofs are elementary linear algebra; their consequence is that no bilinear-form generalization of the inner product can reduce the hull dimension by more than one, which is what eventually drives the optimal SSA speedup of Theorem 3.

3.1 Bilinear Hulls

A bilinear form $B : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ is identified with its matrix in the standard basis via $B(x, y) = xMy^\top$ for a *structure matrix* $M \in \mathbb{F}_q^{n\times n}$; the standard inner product corresponds to $M = I$. All subsequent statements are formulated directly in terms of M.

Definition 2 (M-dual and M-hull). *For a code $C \subseteq \mathbb{F}_q^n$ and a structure matrix $M \in \mathbb{F}_q^{n\times n}$, the* M-dual *and* M-hull *of C are*

$$C^{\perp_M} = \{y \in \mathbb{F}_q^n : xMy^\top = 0 \text{ for all } x \in C\}, \qquad \mathrm{Hull}_M(C) = C \cap C^{\perp_M}.$$

M is nondegenerate *if it is invertible, in which case* $\dim C^{\perp_M} = n - \dim C$.

Remark 2. In this paper, we assume that structure matrices are symmetric. We need this assumption because otherwise the orthogonality would not be symmetric.

Definition 3 (Isometry). *A map $f : \mathbb{F}_q^n \to \mathbb{F}_q^n$ is an* isometry *of B (or of M) if $B(f(x), f(y)) = B(x, y)$ for all $x, y \in \mathbb{F}_q^n$. When $f(x) = xA$ for a matrix $A \in \mathbb{F}_q^{n\times n}$, the isometry condition is equivalent to $AMA^\top = M$.*

For SSA to operate on Hull_M in place of Hull, the M-hull must be invariant under coordinate permutations, i.e.,

$$\mathrm{Hull}_M(\pi(C)) = \pi(\mathrm{Hull}_M(C)) \qquad \text{for all } \pi \in S_n \text{ and all codes } C. \tag{3}$$

3.2 Characterization of Permutation-Invariant Bilinear Hulls

Theorem 1 below characterizes the structure matrices M for which this holds, together with several equivalent reformulations. To this end, we establish the following auxiliary lemmas:

Lemma 1 (Proportionality of quadratic forms). *Let q be odd and $n \geq 3$. If Q_1, Q_2 are nondegenerate quadratic forms on $\mathbb{F}_q^n$ with the same zero set, then $Q_2 = \lambda Q_1$ for some $\lambda \in \mathbb{F}_q^*$.*

Proof. Pick $u_0 \in \mathbb{F}_q^n$ with $Q_1(u_0) \neq 0$ (and hence $Q_2(u_0) \neq 0$, by the equal-zero-set hypothesis); set $\lambda := Q_2(u_0)/Q_1(u_0) \in \mathbb{F}_q^*$. For any nonzero isotropic vector w, the line $\{u_0 + tw : t \in \mathbb{F}_q\}$ yields polynomials

$$f_i(t) := Q_i(u_0 + tw) = Q_i(u_0) + 2tB_i(u_0, w),$$

where B_i denotes the symmetric bilinear form polarized from Q_i. Both f_1, f_2 are linear in t, with the same zero set in $\mathbb{F}_q$ (since Q_1 and Q_2 have the same zero set). Equating root structures forces $B_2(u_0, w) = \lambda B_1(u_0, w)$: if $B_1(u_0, w) = 0$ then $f_1 \equiv Q_1(u_0)$ has empty zero set, so $B_2(u_0, w) = 0$ as well; otherwise f_1 has the unique root $-Q_1(u_0)/(2B_1(u_0, w))$, and equating with the unique root of f_2 gives the desired identity.

For $n \geq 3$ and q odd, the isotropic vectors of any nondegenerate quadratic form Q span $\mathbb{F}_q^n$. By the Witt decomposition, $\mathbb{F}_q^n = H \perp W$ with H an orthogonal sum of hyperbolic planes and W anisotropic of dimension ≤ 2: the hyperbolic bases of H are isotropic and span H; for any $w \in W \setminus \{0\}$, surjectivity of the form on a hyperbolic plane provides $h \in H$ with $Q(h) = -Q(w)$, so $h + w$ is isotropic with nonzero W-component. Combining these vectors yields a spanning set of $\mathbb{F}_q^n$ consisting of isotropic vectors. Hence the linear functional $v \mapsto B_2(u_0, v) - \lambda B_1(u_0, v)$ vanishes on a spanning set, so $B_2(u_0, \cdot) = \lambda B_1(u_0, \cdot)$ identically.

Repeating the argument with any $u_0' \in \mathbb{F}_q^n$ satisfying $Q_1(u_0') \neq 0$ gives $B_2(u_0', \cdot) = \lambda_{u_0'} B_1(u_0', \cdot)$ for some $\lambda_{u_0'} \in \mathbb{F}_q^*$. Choosing u_0' so that $B_1(u_0, u_0') \neq 0$ (possible by non-degeneracy of B_1 together with $\dim\{v : B_1(u_0, v) \neq 0\} = n-1$), the symmetry of the bilinear forms yields

$$\lambda B_1(u_0, u_0') = B_2(u_0, u_0') = B_2(u_0', u_0) = \lambda_{u_0'} B_1(u_0', u_0) = \lambda_{u_0'} B_1(u_0, u_0'),$$

so $\lambda_{u_0'} = \lambda$. Since $\{u : Q_1(u) \neq 0\}$ spans $\mathbb{F}_q^n$, $B_2 = \lambda B_1$ identically, hence $Q_2 = \lambda Q_1$.

Lemma 2. *Let $n \geq 3$ and let $N \neq 0$ be a symmetric matrix over $\mathbb{F}_q$ such that, for every $P \in S_n$, one has $PNP^\top = \nu_P N$ for some $\nu_P \in \mathbb{F}_q^*$. Then $\nu_P = 1$ for all P, and hence $PNP^\top = N$ for every $P \in S_n$.*

Proof. As $N \neq 0$, each ν_P is uniquely determined, and comparing $(P_1P_2)N(P_1P_2)^\top = \nu_{P_1P_2}N$ with $P_1(P_2NP_2^\top)P_1^\top = \nu_{P_1}\nu_{P_2}N$ shows $P \mapsto \nu_P$ is a homomorphism $S_n \to \mathbb{F}_q^*$; in particular $\nu_\tau^2 = 1$ for every transposition τ. If q is even, then $|\mathbb{F}_q^*| = q - 1$ is odd, so $\nu_\tau = 1$. If q is odd and $\nu_\tau = -1$ for some $\tau = (i\,j)$, then $\tau N \tau^\top = -N$ reads $N_{\tau(k)\tau(l)} = -N_{kl}$: the (i, j) entry with $N_{ji} = N_{ij}$ gives $N_{ij} = 0$ for all $i \neq j$, and the (i, i) entry gives $N_{jj} = -N_{ii}$, so (as $n \geq 3$) every $N_{ii} = 0$; thus $N = 0$, contradicting the assumption $N \neq 0$. Hence $\nu_\tau = 1$ in either case, and since transpositions generate S_n, $\nu_P \equiv 1$.

Theorem 1. *Let B be a bilinear form on $\mathbb{F}_q^n$ with symmetric nondegenerate structure matrix $M \in \mathbb{F}_q^{n \times n}$, and assume $n \geq 3$. The following are equivalent.*

(i) Every coordinate permutation $\pi \in S_n$ is an isometry of B:

$$B\big(\pi(x),\pi(y)\big) = B(x,y) \quad \textit{for all } x,y \in \mathbb{F}_q^n.$$

(ii) $M = aI+bJ$ for some $a,b \in \mathbb{F}_q$, where I is the identity matrix and $J = \mathbf{1}\mathbf{1}^\top$ is the all-ones matrix.

(iii) For every $\pi \in S_n$ and every code $C \subseteq \mathbb{F}_q^n$,

$$\pi\big(C^{\perp_M}\big) = \big(\pi(C)\big)^{\perp_M}.$$

(iv) For every $\pi \in S_n$ and every code $C \subseteq \mathbb{F}_q^n$,

$$\pi\big(\mathrm{Hull}_M(C)\big) = \mathrm{Hull}_M\big(\pi(C)\big).$$

Proof. Let $P = P_\pi$ denote the permutation matrix of π, so $\pi(x) = xP$.

(i) $\Rightarrow$ (ii): The isometry condition $(xP)M(yP)^\top = xMy^\top$ for all $x,y \in \mathbb{F}_q^n$ rewrites as $x(PMP^\top)y^\top = xMy^\top$, hence $PMP^\top = M$. Entry-wise, $(PMP^\top)_{ij} = M_{\pi(i),\pi(j)}$, so

$$M_{\pi(i),\pi(j)} = M_{ij} \quad \text{for all } \pi \in S_n \text{ and all } i,j \in [n].$$

For $n \geq 2$, the diagonal action of S_n on $[n]\times[n]$ has exactly two orbits, $\{(i,i) : i \in [n]\}$ and $\{(i,j) : i \neq j\}$, so M is constant on each: $M_{ii} = \alpha$ and $M_{ij} = \beta$ $(i \neq j)$ for some $\alpha,\beta \in \mathbb{F}_q$, giving $M = (\alpha-\beta)I + \beta J$.

(ii) $\Rightarrow$ (iii): For any code C and any $y \in \mathbb{F}_q^n$, using $(yP^\top)^\top = Py^\top$,

$$\begin{aligned} y \in \big(\pi(C)\big)^{\perp_M} &\iff (xP)My^\top = x(PM)y^\top = 0 \quad \text{for all } x \in C, \\ y \in \pi\big(C^{\perp_M}\big) &\iff xM(yP^\top)^\top = x(MP)y^\top = 0 \quad \text{for all } x \in C. \end{aligned}$$

Condition (ii), i.e. $M = aI + bJ$, gives $PM = MP$ (as J commutes with every permutation matrix), so the two conditions coincide.

(iii) $\Rightarrow$ (iv): For any C and π,

$$\pi\big(\mathrm{Hull}_M(C)\big) = \pi(C) \cap \pi\big(C^{\perp_M}\big), \qquad \mathrm{Hull}_M\big(\pi(C)\big) = \pi(C) \cap \big(\pi(C)\big)^{\perp_M}.$$

Hence $\pi(C^{\perp_M}) = (\pi(C))^{\perp_M}$ implies $\pi(\mathrm{Hull}_M(C)) = \mathrm{Hull}_M(\pi(C))$.

(iv) $\Rightarrow$ (i): Fix $P = P_\pi$ and write $M' = PMP^\top$, which is again symmetric and nondegenerate. Since $(cP)M(zP)^\top = c(PMP^\top)z^\top = cM'z^\top$ for all c,z, we have $(\pi C)^{\perp_M} = \pi\big(C^{\perp_{M'}}\big)$, hence $\mathrm{Hull}_M(\pi C) = \pi\big(\mathrm{Hull}_{M'}(C)\big)$; comparing with (iv) yields

$$\mathrm{Hull}_M(C) = \mathrm{Hull}_{M'}(C) \qquad \text{for every code } C \subseteq \mathbb{F}_q^n. \tag{4}$$

Taking $C = \langle x\rangle$ and using $\mathrm{Hull}_M(\langle x\rangle) = \langle x\rangle$ iff $xMx^\top = 0$ (and $\{0\}$ otherwise), M and M' have the same isotropic set:

$$xMx^\top = 0 \iff xM'x^\top = 0 \qquad (x \in \mathbb{F}_q^n). \tag{5}$$

Case q odd. By (5) the quadratic forms M and M' share their zero set, so $M' = \lambda_P M$ for some $\lambda_P \in \mathbb{F}_q^*$ by Lemma 1. As this holds for every P, Lemma 2 applied to $N = M$ (nonzero, as M is nondegenerate) gives $PMP^\top = M$.

Case q even. Here $xMx^\top = \sum_i M_{ii}x_i^2 = (d{\cdot}x)^2$ with $d = (\sqrt{M_{11}}, \ldots, \sqrt{M_{nn}})$, so by (5) both isotropic sets equal $Z = \{x : d \cdot x = 0\}$. For $x \in Z$ and any y one has $x \in \mathrm{Hull}_M(\langle x, y\rangle) \iff xMy^\top = 0$, and likewise for M'; since $\mathrm{Hull}_M(\langle x, y\rangle) = \mathrm{Hull}_{M'}(\langle x, y\rangle)$ by (4), the forms $y \mapsto xMy^\top$ and $y \mapsto xM'y^\top$ share a kernel, so $xM' = \mu(x)\, xM$ with $\mu(x) \in \mathbb{F}_q^*$. We claim μ is constant on $Z \setminus \{0\}$. For a scalar $c \in \mathbb{F}_q^*$, $(cx)M' = c\,(xM') = c\,\mu(x)\, xM = \mu(x)\,(cx)M$, so $\mu(cx) = \mu(x)$. For linearly independent $x, y \in Z$ we have $x + y \in Z$ (as Z is a subspace), hence

$$\mu(x+y)\,(xM + yM) = (x+y)M' = xM' + yM' = \mu(x)\, xM + \mu(y)\, yM,$$

that is, $\big(\mu(x+y) - \mu(x)\big)xM + \big(\mu(x+y) - \mu(y)\big)yM = 0$. Since M is invertible and x, y are independent, xM and yM are independent, so $\mu(x+y) = \mu(x) = \mu(y)$. Thus any two nonzero vectors of Z have the same μ-value (directly if independent, via a scalar multiple otherwise), proving the claim; write $\mu \equiv \lambda_P$. Then $x(M' - \lambda_P M) = 0$ for all $x \in Z$. If $d = 0$ then $Z = \mathbb{F}_q^n$ and $M' = \lambda_P M$. By Lemma 2, $PMP^\top = M$ follows and we are done. Therefore, we assume $d \neq 0$, which makes $Z = d^\perp$ a hyperplane, and $M' - \lambda_P M$ is symmetric of rank ≤ 1 with left kernel containing $d^\perp$, so we have

$$PMP^\top = \lambda_P\, M + c_P\, dd^\top \qquad (c_P \in \mathbb{F}_q).$$

Taking (i, i) entries and using $d_i^2 = M_{ii}$ gives $d_{\pi(i)}^2 = (\lambda_P + c_P)\, d_i^2$; since S_n is transitive, d is either 0 or nowhere zero. Since we have excluded the case $d = 0$ above, we assume that every $d_i \neq 0$. For a transposition $\tau = (a\, b)$ a fixed coordinate $c \notin \{a, b\}$ (which exists as $n \geq 3$) forces $\lambda_\tau + c_\tau = 1$, whence $d_{\tau(i)} = d_i$ for all i and in particular $d_a = d_b$; ranging over transpositions, $d = d_0\mathbf{1}$ and $dd^\top = d_0^2 J$. Now $\lambda_P + c_P = 1$ for every P, so $\widetilde{M} := M + d_0^2 J$ (nonzero, as M is nondegenerate but $d_0^2 J$ is not) satisfies

$$P\widetilde{M}P^\top = \lambda_P M + (c_P + 1)d_0^2 J = \lambda_P \widetilde{M},$$

and Lemma 2 gives $P\widetilde{M}P^\top = \widetilde{M}$, which implies $PMP^\top = M$. This completes the cycle (i) $\Rightarrow$ (ii) $\Rightarrow$ (iii) $\Rightarrow$ (iv) $\Rightarrow$ (i). □

Remark 3 (Optimality of the hypotheses). Both the bound $n \geq 3$ and the symmetry of M are necessary in Theorem 1.

$n \geq 3$ is needed, even for symmetric M. Over $\mathbb{F}_5$ take the symmetric nondegenerate matrix $M = \mathrm{diag}(1, 2)$. Its quadratic form $x_1^2 + 2x_2^2$ is anisotropic, since -2 is a non-square in $\mathbb{F}_5$; hence $\mathrm{Hull}_M(C) = \{0\}$ for every code $C \subseteq \mathbb{F}_5^2$, and condition (iv) holds trivially. Yet $M \neq aI + bJ$, so (i) fails, and the equivalence breaks down at $n = 2$.

Symmetry is needed at $n = 3$. Over $\mathbb{F}_3$ the non-symmetric nondegenerate matrix $M = \left(\begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 2 \end{smallmatrix}\right)$ has permutation-invariant Hull_M (condition (iv)) while

$M \neq aI + bJ$. Such asymmetric counterexamples have degenerate symmetric part and rely on the fact that, for $n = 3$, no transposition is disjoint from a given pair of coordinates; we know of none for $n \geq 4$. Whether the symmetry hypothesis can be dropped for $n \geq 4$ is left as future work.

The crucial equivalence for the cryptanalytic application is (ii) $\Leftrightarrow$ (iv): a structure matrix yields a permutation-invariant hull, and is thus usable as an SSA invariant in place of the standard hull, if and only if it has the form $M = aI+bJ$.

3.3 Hull Dimension Bound

Theorem 1 identifies $aI + bJ$ as the only family of structure matrices yielding SSA-compatible hulls. We now show that, for any such M, the resulting M-hull has dimension within one of the standard hull. The argument rests on the following standard rank-perturbation bound, included for self-containment.

Lemma 3 (Rank perturbation). *For any matrices $A, B \in \mathbb{F}_q^{m\times n}$,*

$$|\operatorname{rank}(A + B) - \operatorname{rank} A| \leq \operatorname{rank} B.$$

Proof. Set $C = A + B$. Since $\operatorname{Im}(C) \subseteq \operatorname{Im}(A) + \operatorname{Im}(B)$,

$$\operatorname{rank} C \leq \dim\big(\operatorname{Im}(A) + \operatorname{Im}(B)\big) \leq \operatorname{rank} A + \operatorname{rank} B,$$

so $\operatorname{rank} C - \operatorname{rank} A \leq \operatorname{rank} B$. Applying the same inequality to $A = C - B$ gives $\operatorname{rank} A - \operatorname{rank} C \leq \operatorname{rank} B$, completing the proof. □

Theorem 2 (Dimension bound). *Let C be an $[n, k]_q$ code with generator matrix $G \in \mathbb{F}_q^{k\times n}$, and let $M = aI + bJ$ with $a \neq 0$. (Nondegeneracy of M is equivalent to $a \neq 0$ and $a + nb \neq 0$; we do not require it here.) Then*

$$|\dim \operatorname{Hull}_M(C) - \dim \operatorname{Hull}(C)| \leq 1.$$

Proof. A vector $x = uG$ lies in $\operatorname{Hull}_M(C) = C \cap C^{\perp_M}$ if and only if $xMG^\top = 0$, i.e., $u\,(GMG^\top) = 0$. Hence

$$\dim \operatorname{Hull}_M(C) = k - \operatorname{rank}(GMG^\top), \tag{6}$$

and likewise $\dim \operatorname{Hull}(C) = k - \operatorname{rank}(GG^\top)$.

Expanding with $M = aI + bJ$ and $J = \mathbf{1}\mathbf{1}^\top$,

$$GMG^\top = aGG^\top + bGJG^\top = aGG^\top + bvv^\top, \qquad v = G\mathbf{1}^\top \in \mathbb{F}_q^k,$$

where $GJG^\top = G\mathbf{1}\mathbf{1}^\top G^\top = vv^\top$. Since $a \neq 0$, $\operatorname{rank}(aGG^\top) = \operatorname{rank}(GG^\top)$, and $\operatorname{rank}(bvv^\top) \leq 1$. By Lemma 3,

$$\begin{aligned}\big|\operatorname{rank}(GMG^\top) - \operatorname{rank}(GG^\top)\big| &= \big|\operatorname{rank}(aGG^\top + bvv^\top) - \operatorname{rank}(aGG^\top)\big| \\ &\leq \operatorname{rank}(bvv^\top) \leq 1.\end{aligned}$$

Combining with (6) yields the claim. □

3.4 Self-Dual Reduction

For self-dual codes, which are the relevant case for SPECK, the dimension reduction is not merely bounded but is achieved exactly, and any nondegenerate $M = aI + bJ$ with $b \neq 0$ realizes it.

Corollary 1. *Let C be a self-dual $[n, n/2]_q$ code and let $M = aI + bJ$ with $a \neq 0$ and $b \neq 0$. If $n \not\equiv 0 \pmod{p}$, where $p = \mathrm{char}(\mathbb{F}_q)$, then*

$$\dim \mathrm{Hull}_M(C) = \frac{n}{2} - 1.$$

The matrix M is additionally nondegenerate iff $a + nb \neq 0$, a condition independent of the dimension formula above; the canonical choice $a = b = 1$ satisfies this whenever $n + 1 \not\equiv 0 \pmod{p}$, in particular at SPECK parameters.

Proof. By self-duality, $C = C^{\perp}$, so $GG^{\top} = 0$. Continuing the calculation of Theorem 2,

$$GMG^{\top} = aGG^{\top} + bvv^{\top} = bvv^{\top}, \qquad v = G\mathbf{1}^{\top}.$$

We claim $v \neq 0$ under the hypothesis $p \nmid n$. Indeed, $v = G\mathbf{1}^{\top} = 0$ is equivalent to $\mathbf{1} \in C^{\perp} = C$, which forces $\langle \mathbf{1}, \mathbf{1} \rangle = n \cdot 1_{\mathbb{F}_q} = 0$, contradicting $p \nmid n$. Since $b \neq 0$ and $v \neq 0$, $\mathrm{rank}(bvv^{\top}) = 1$, and (6) gives

$$\dim \mathrm{Hull}_M(C) = k - \mathrm{rank}(GMG^{\top}) = \frac{n}{2} - 1.$$

□

For SPECK's parameters ($n = 252$, with $p = 127$ in the Low q regime and $p = 8861$ in the High q regime), $p \nmid 252$ in both cases, so Corollary 1 applies to every self-dual code used in SPECK. For the canonical choice $M = I + J$, $a + nb = 253$ which is nonzero in both $\mathbb{F}_{127}$ ($253 \equiv -1$) and $\mathbb{F}_{8861}$ ($253 \neq 0$), so M is nondegenerate and the bilinear hull always has dimension $n/2 - 1 = 125$, one less than the standard hull of dimension 126.

4 Application to SPECK: Factor-q SSA Cost Reduction on PEP Instances

We now apply the structural results of Section 3 to SPECK [3], the only current code-based signature candidate built directly on PEP. The output is a factor-q reduction in the cost of SSA on the underlying PEP instances; we emphasize that SSA is not necessarily the best known attack on SPECK as a scheme, and the reduction below is a statement about SSA's running time, not about SPECK's overall security level. Section 4.1 introduces SPECK and the PECK problem. Section 4.2 reviews when PECK reduces to PEP. Sections 4.3 to 4.5 present the bilinear-hull SSA, its cost, and its scope. Section 4.6 reports the concrete numerical impact on SSA at SPECK's recommended parameters.

4.1 The SPECK Signature Scheme

SPECK [3] is a code-based digital signature scheme whose security relies on the hardness of a new problem called PECK (Permutation Equivalence of Codes and Kernel). It is inspired by the permutation version of LESS [8] but achieves faster verification by replacing the expensive systematic form computation with simple vector-matrix multiplications.

We now introduce the interactive SPECK protocol, which can be transformed into a signature scheme via the Fiat-Shamir transform.

Key generation. A random k-dimensional self-orthogonal code $C \subseteq \mathbb{F}_q^n$ is sampled, with generator matrix $G \in \mathbb{F}_q^{k \times n}$ and parity-check matrix $H \in \mathbb{F}_q^{(n-k) \times n}$. At SPECK's parameter setting $k = n/2$, self-orthogonality forces $C = C^\perp$, so C is in fact self-dual, and we use the two terms interchangeably below. A secret permutation $P \in S_n$ and a random invertible matrix $S \in GL_{n-k}(\mathbb{F}_q)$ are chosen. The public key is (G, H') where $H' = SHP$, and the private key is P.

Commitment. The prover samples a random $(u, P^*) \xleftarrow{\$} \mathbb{F}_q^k \times S_n$ via a random oracle and seed, and computes $c = uGP^*$. The commitment is defined by $\mathsf{cmt} = \mathsf{Hash}(\mathsf{LexMin}(c))$, where LexMin maps c to the lexicographically minimal element of its equivalence class under S_n.

Response. The verifier chooses a random bit $b \in \{0, 1\}$, and sends it to the prover.

The prover generates a response rsp as follows:

- If $b = 0$: the prover defines the response rsp by $\mathsf{rsp} := \mathsf{seed}$;
- If $b = 1$: the prover computes $y = cP^{*-1}P$, and lets $\mathsf{rsp} := y$.

Verification.

- If $b = 0$: the verifier samples (u, P^*) by using seed, and checks whether cmt is equal to $\mathsf{Hash}(\mathsf{LexMin}(uGP^*))$;
- If $b = 1$: the verifier checks whether $yH'^\top = \mathbf{0}$ and $\mathsf{cmt} = \mathsf{Hash}(\mathsf{LexMin}(y))$.

The PECK problem. Forging a SPECK signature reduces to solving an instance of the PECK problem. We recall the formal definitions from [3]; the distributions below capture how the keys and commitments used in SPECK are sampled.

Definition 4 ($\mathcal{D}^{\mathrm{PEP}}_{q,n,k}$ [3, Def. 3]). *The distribution $\mathcal{D}^{\mathrm{PEP}}_{q,n,k}$ samples:*

1. *a uniformly random k-dimensional self-orthogonal code $C \subseteq \mathbb{F}_q^n$ with generator matrix G and parity-check matrix H;*
2. *a uniformly random invertible $S \xleftarrow{\$} GL_{n-k}(\mathbb{F}_q)$ and permutation $P \xleftarrow{\$} S_n$;*

and outputs $\{G, H'\}$ where $H' = SHP$.

Definition 5 ($\mathcal{D}^{\mathrm{PECK}}_{q,n,k}$ **[3, Def. 4]).** *The distribution* $\mathcal{D}^{\mathrm{PECK}}_{q,n,k}$ *draws* $\{G, H'\} \leftarrow \mathcal{D}^{\mathrm{PEP}}_{q,n,k}$*, samples* $u \xleftarrow{\$} \mathbb{F}_q^k$*, computes* $c = uG$*, and returns* $\{c, G, H'\}$*.*

We denote by $\mathrm{PEP}_{q,n,k}$ the search problem of Definition 1 instantiated on inputs drawn from $\mathcal{D}^{\mathrm{PEP}}_{q,n,k}$: given $\{G, H'\} \leftarrow \mathcal{D}^{\mathrm{PEP}}_{q,n,k}$, find $P \in S_n$ such that $GPH'^{\top} = \mathbf{0}$ [3, Problem 4]. The distinction with Definition 1 is solely the uniform-random sampling of (G, S, P), which is what turns the problem into a cryptographic assumption.

Definition 6 (Permutation Equivalence of Codes and Kernel $\mathrm{PECK}_{q,n,k}$ **[3, Problem 5]).** *Given* $\{c, G, H'\} \leftarrow \mathcal{D}^{\mathrm{PECK}}_{q,n,k}$*, find* $P \in S_n$ *such that* $cPH'^{\top} = \mathbf{0}$*.*

Note that $\mathrm{PECK}_{q,n,k}$ asks for a permutation sending a *specific* random codeword c of C into C', whereas $\mathrm{PEP}_{q,n,k}$ asks for a permutation sending the *entire* code C to C'. The exact relationship between these two problems, and when a PEP solver suffices to solve PECK, is reviewed in Section 4.2 as the starting point of our reduction.

Recommended parameters. SPECK recommends two parameter regimes with identical code parameters $n = 252$, $k = 126$: the Low q regime with $q = 127$ and the High q regime with $q = 8861$, corresponding to different attack landscapes analyzed in Section 4.2. Within each regime, several Fiat–Shamir parameter choices (t, w) are offered trading signature size for latency. Our SSA-based analysis depends only on (n, k, q) and is insensitive to (t, w), so we refer the reader to [3, Table 1] for the full list of recommended instances and their signature sizes.

4.2 From PECK to PEP: the High-q Regime

The following relationships between PECK, PEP, and PKP are established in [3, Section 6].

Proposition 1 ([3, Section 6.1]). *For all* (q, n, k)*:*

1. $\mathrm{PECK}_{q,n,k}$ *is at most as hard as* $\mathrm{PEP}_{q,n,k}$*: given a PECK instance* $\{c, G, H'\}$*, one can invoke a PEP solver on* $\{G, H'\}$ *to obtain* P *sending* C *to* C'*, which necessarily also sends* c *to* C'*.*
2. $\mathrm{PECK}_{q,n,k}$ *is at most as hard as* $\mathrm{PKP}_{q,n,k}$*: the PECK problem can be viewed as a variant of the Permuted Kernel Problem with repeated entries.*

The converse direction, whether $\mathrm{PEP}_{q,n,k}$ reduces to $\mathrm{PECK}_{q,n,k}$, depends on the number of solutions.

Proposition 2 (Expected number of solutions [3, Prop. 2]). *Let* $\{c, G, H'\} \leftarrow \mathcal{D}^{\mathrm{PECK}}_{q,n,k}$ *with* $m = \psi(c)$ *being the multiplicity vector of* c*. The expected number of solutions is*

$$N_{\mathrm{sol}} = 1 + \frac{|\mathbb{A}_n(m)| - 1}{q^{n-k}},$$

where $\mathbb{A}_n(m)$ *denotes the set of arrangements of* c *on* n *positions.*

Algorithm 1: Bilinear-hull SSA on a PECK instance (High q regime)

Input: PECK instance $\{c, G, H'\}$ in the High q regime
Output: Permutation $P \in S_n$ such that $cPH'^{\top} = \mathbf{0}$
`// Stage 1: Bilinear hull preprocessing (using` $M = I + J$`)`
$v \leftarrow G\mathbf{1}^{\top}$;
$W \leftarrow vv^{\top}$; `//` $= GMG^{\top}$ `since` $GG^{\top} = 0$ `for self-dual` C
Compute a basis $\{c_1, \ldots, c_{k-1}\}$ of $\ker(W) \subseteq \mathbb{F}_q^k$;
Basis of $\mathrm{Hull}_M(C)$: $\{c_i G\}_{i=1}^{k-1}$; `//` `dimension` $n/2 - 1$
Compute $\mathrm{Hull}_M(C')$ analogously from a generator matrix of C';
`// Stage 2: SSA on reduced hull`
Run SSA on input $(\mathrm{Hull}_M(C), \mathrm{Hull}_M(C'))$ to obtain $P \in S_n$;
return P

When $N_{\mathrm{sol}} \approx 1$, the PEP solution is the unique PECK solution, so the two problems are equivalent.

Definition 7 (High q and Low q regimes [3, Section 6.1]). *Let $R = k/n$ denote the code rate.*

- *High q regime: $q > (n/e)^{1/(1-R)}$. Here $N_{\mathrm{sol}} \approx 1$ with overwhelming probability, so $\mathrm{PECK}_{q,n,k}$ is computationally equivalent to $\mathrm{PEP}_{q,n,k}$.*
- *Low q regime: $q \ll (n/e)^{1/(1-R)}$. Here $N_{\mathrm{sol}} \gg 1$ and solving PEP alone does not identify the correct permutation for PECK. PKP-based collision search solvers [17] are more effective in this regime.*

For SPECK's recommended parameters with rate $R = 1/2$ and $n = 252$, the threshold is $q > (252/e)^2 \approx 8595$. SPECK uses $q = 8861$ for the High q regime and $q = 127$ for the Low q regime.

4.3 Bilinear-Hull SSA on a PECK Instance

The procedure proceeds in two stages. First, we reduce a PECK instance to a PEP instance using Proposition 1(1). Second, we solve the PEP instance using SSA with the bilinear hull in place of the standard hull.

The key observation is that replacing $\mathrm{Hull}(C)$ with $\mathrm{Hull}_M(C)$ in the SSA is valid: by Theorem 1, the bilinear hull with $M = aI + bJ$ satisfies $\mathrm{Hull}_M(\pi(C)) = \pi(\mathrm{Hull}_M(C))$, so the permutation invariance required by the SSA is preserved. By Corollary 1, this hull has dimension $n/2 - 1$ for any nondegenerate $M = aI + bJ$ with $b \neq 0$ on the self-dual codes used in SPECK, compared to $n/2$ for the standard hull.

Algorithm 1 describes the full procedure. In Stage 1, we exploit the self-duality of SPECK codes: since $GG^{\top} = 0$, the matrix $GMG^{\top}$ simplifies to $bvv^{\top}$ (a rank-1 matrix), and hence $\mathrm{Hull}_M(C) = \{uG : u \in \ker(vv^{\top})\}$ has a basis computable from a single kernel computation. The preprocessing cost is $O(k^2)$ field operations, which is negligible compared to SSA itself. In Stage 2, the standard SSA runs on codes of hull dimension $n/2 - 1$ instead of $n/2$.

Remark 4. To compute $\mathrm{Hull}_M(C')$ from the parity-check matrix H', observe that $C' = \ker(H'^{\top})$ and $\mathrm{Hull}_M(C') = \pi(\mathrm{Hull}_M(C))$. One can either compute a generator matrix G' of C' from H' (at cost $O(n^3)$, which is already incurred in SSA) or use the characterization $\mathrm{Hull}_M(C') = \{y \in C' : yMG'^{\top} = 0\}$ directly.

4.4 Cost Analysis

Theorem 3 (Main result). *Let SPECK use a self-dual $[n, n/2]_q$ code with $n \not\equiv 0 \pmod p$. In the High q regime ($q > (n/e)^2$), the bilinear-hull SSA solves* $\mathrm{PECK}_{q,n,k}$ *in time*

$$O(n^3 + n^2 q^{n/2-1} \log n),$$

a factor of q below the standard SSA cost $O(n^3 + n^2 q^{n/2} \log n)$ on the same instance.

Proof. By Proposition 1(1), solving $\mathrm{PEP}_{q,n,k}$ solves $\mathrm{PECK}_{q,n,k}$. In the High q regime, $N_{\mathrm{sol}} \approx 1$ by Proposition 2, so the PEP solution is the unique PECK solution. By Corollary 1, $\dim \mathrm{Hull}_M(C) = n/2-1$ for any nondegenerate $M = aI+bJ$ with $b \neq 0$, and this hull is permutation-invariant by Theorem 1. Substituting $h = n/2 - 1$ into the SSA cost (2) gives the claimed bound. The bilinear-hull preprocessing (Algorithm 1, Stage 1) costs $O(k^2) = O(n^2)$, which is absorbed by the $O(n^3)$ term. □

Corollary 2. *In the High q regime, the bilinear-hull SSA on* $\mathrm{PECK}_{q,n,k}$ *has* $\log_2$*-cost*

$$\log_2(q^{n/2-1}) = \frac{n}{2} \log_2 q - \log_2 q,$$

i.e., a reduction of $\Delta = \log_2 q$ in the dominant exponent compared with standard SSA.

Proof. The ratio of the old and new dominant terms is $q^{n/2}/q^{n/2-1} = q$. □

For SPECK's High q parameter $q = 8861$, this gives $\Delta = \log_2(8861) \approx 13.11$. For the Low q parameter $q = 127$, the bilinear hull still reduces the PEP-solving cost by $\log_2(127) \approx 6.99$, but this does not directly translate to a PECK speedup because the PEP-to-PECK reduction fails in the Low q regime (Definition 7).

4.5 Applicability and Limitations

We summarize the scope of the bilinear-hull SSA on SPECK.

High q regime. The bilinear-hull SSA solves both PEP and PECK with a factor-q cost reduction over standard SSA. For SPECK's $q = 8861$ this amounts to $\Delta \approx 13$ in $\log_2$-cost. SSA is one of several known PEP solvers (cf. [3, Section 6.1]) and is not necessarily the cheapest; we therefore make no claim about the bit-security of SPECK as a scheme, only about SSA's running time on the underlying PEP instance.

Table 1. $\log_2$-cost of SSA on the underlying PEP instance for SPECK ($n = 252$, $k = 126$). The column $\log_2 \mathsf{C}_{\mathrm{SSA}}$ is the standard SSA cost ($= \frac{n}{2}\log_2 q$), $\log_2 \mathsf{C}^B_{\mathrm{SSA}}$ is the bilinear-hull SSA cost ($= (\frac{n}{2}-1)\log_2 q$), and $\Delta = \log_2 q$ is the cost reduction in $\log_2$-form. For reference, the last column lists the cost of the best known attack on SPECK as a scheme, which need not equal the SSA cost on PEP.

Instance	q	$\log_2 \mathsf{C}_{\mathrm{SSA}}$	$\log_2 \mathsf{C}^B_{\mathrm{SSA}}$	Δ	Best known on SPECK
Speck-Low	127	880.6	873.6	7.0	$\approx 2^{130}$ [3]
Speck-High	8861	1652.3	1639.2	13.1	exponential in n [3, Sec. 6.1]

Low q regime. The bilinear hull reduces the PEP-solving cost by a factor of $q = 127$, but this does not translate to a faster PECK solver because $N_{\mathrm{sol}} \gg 1$. In this regime, the PKP-based collision search of [3, Section 6.2] is more effective than SSA, and that path is unaffected by our technique. Hybrid strategies that enumerate the (multiple) PEP solutions and then filter by the PECK constraint $cPH'^{\top} = \mathbf{0}$ could benefit from the reduced PEP-solving cost; we leave a precise analysis of such hybrids to future work.

Tightness within the bilinear framework. The cost bound in Theorem 3 is tight in the following sense: it cannot be improved within any bilinear-form generalization of the hull. By Theorem 1, the only M whose Hull_M is permutation-invariant is $M = aI + bJ$; by Theorem 2, every such M reduces the hull dimension by at most one. Hence the SSA cost cannot be reduced by more than a factor of q through any bilinear-form generalization of the inner product. Any further reduction must come from invariants outside the bilinear framework, such as higher-order tensor invariants or non-bilinear algebraic structures.

4.6 Concrete SSA Cost on SPECK Instances

We tabulate the concrete cost of standard and bilinear-hull SSA on SPECK's recommended instances. The SSA cost on an $[n,k]_q$ code with hull dimension h is dominated by q^h (see (2)); we report it in $\log_2$-form, $\log_2(q^h) = h\log_2 q$. Table 1 compares the two SSA variants for both SPECK regimes.

Two observations emerge.

SSA is not the bottleneck. For both regimes, the bilinear-hull SSA cost on PEP exceeds 2^{870}, so our factor-q improvement does not change the practical assessment of SSA on PEP at SPECK's recommended parameters. The best known attack on SPECK in the Low q regime is the PKP-based collision search with cost $\approx 2^{130}$ [3], which does not use the hull and is unaffected by our technique. In the High q regime, security relies on the hardness of PEP, but other PEP solvers exist (e.g., the low-weight-codeword and meet-in-the-middle attacks discussed in [3, Section 6.1]); the cheapest known attack on SPECK as a scheme need not be SSA, so we make no claim on SPECK's bit-security.

Implications for parameter design. The bilinear-hull SSA gives the tight cost of SSA on PEP-based instances and is therefore relevant whenever future PEP-based proposals shrink n or q for efficiency. Concretely, the minimum hull dimension h needed for the SSA cost to exceed 2^λ satisfies

$$h \geq \lceil \lambda / \log_2 q \rceil + 1$$

rather than $\lceil \lambda / \log_2 q \rceil$. For SPECK's currently recommended (n, q) this correction is invisible against the dominant non-SSA attacks, but it becomes meaningful for any future PEP-based variant whose $\log_2 \mathsf{C}_{\mathrm{SSA}}$ is the bottleneck.

5 Discussion on Related Schemes

We examine the applicability of the bilinear hull technique to other code-based signature schemes.

5.1 LESS: Inapplicability Due to LEP vs. PEP

The LESS signature scheme [8] is based on the *Linear Equivalence Problem* (LEP), which asks to find a *monomial transformation* $\mu = D \cdot \pi$ (a permutation composed with a diagonal scaling matrix) mapping one code onto another. This is a strictly more general notion than permutation equivalence. The bilinear hull technique does not directly apply to LESS for the following reason.

Proposition 3. *Let $M = aI + bJ$ be a nondegenerate structure matrix with $b \neq 0$. Then* Hull_M *is* not *invariant under monomial transformations in general. That is, there exist a code C and a monomial transformation $\mu = D \cdot \pi$ such that* $\mathrm{Hull}_M(\mu(C)) \neq \mu(\mathrm{Hull}_M(C))$.

Proof. We give an explicit counterexample, then explain the structural reason.

Counterexample. Take $n = 3$, $q = 5$, $M = I + J$, and the code $C = \langle (1, 2, 3) \rangle \subseteq \mathbb{F}_5^3$. Using the identity $xMx^\top = (\sum_i x_i)^2 + \sum_i x_i^2$ (valid for $a = b = 1$),

$$(1, 2, 3)\, M\, (1, 2, 3)^\top = 6^2 + 14 = 50 \equiv 0 \pmod 5,$$

so C is M-self-orthogonal and $\mathrm{Hull}_M(C) = C$ has dimension 1. Let $\mu = D$ with $D = \mathrm{diag}(2, 1, 1)$ (and $\pi = \mathrm{id}$); then $\mu(C) = \langle (2, 2, 3) \rangle$. A direct computation gives

$$(2, 2, 3)\, M\, (2, 2, 3)^\top = 7^2 + 17 = 66 \equiv 1 \not\equiv 0 \pmod 5,$$

so $\mu(C)$ is not M-self-orthogonal and hence $\mathrm{Hull}_M(\mu(C)) = \{0\}$. Therefore $\mu(\mathrm{Hull}_M(C)) = \mu(C) \neq \{0\} = \mathrm{Hull}_M(\mu(C))$.

Structural reason. For a monomial $\mu = D \cdot \pi$ with corresponding matrix DP,

$$(DP)M(DP)^\top = D(PMP^\top)D^\top = D(aI + bJ)D^\top = aD^2 + bDJD^\top,$$

where $D^2 = \mathrm{diag}(d_1^2, \ldots, d_n^2)$ and $(DJD^\top)_{ij} = d_i d_j$. This equals $aI + bJ$ only when $D = \alpha I$ for some scalar α (uniform scaling); for non-uniform D, the bilinear

form is not preserved by μ. By the same argument as Theorem 1 (i)$\Leftrightarrow$(iv) applied to monomial transformations in place of permutations, this loss of M-isometry yields, for some code C, a witness to the failure of hull invariance under μ, as exemplified by the construction above.

Remark 5. An indirect route exists via the *closure* construction of Sendrier–Simos [15], which reduces LEP to PEP by expanding the code length from n to $n(q-1)$. The bilinear hull technique could in principle be applied to the closure codes. However, this approach is outperformed by the dedicated LEP solvers [7,**?**,**?**], which do not go through PEP and run in time independent of the hull dimension. In addition, for $q \geq 5$ the closure codes are always weakly self-dual, so their hull dimension is near-maximal and SSA is ineffective on them irrespective of the choice of bilinear form.

5.2 PERK: Inapplicability Due to PKP Foundation

The PERK signature scheme [1] is based on the Permuted Kernel Problem (PKP), which asks to find a permutation mapping a given vector into the kernel of a given matrix. Unlike PEP, PKP does not involve comparing two codes and does not use hulls. The SSA and the bilinear hull technique are therefore entirely inapplicable to PERK.

5.3 Scope of the Bilinear Hull Technique

Summarizing, the bilinear-hull technique applies to signature schemes whose security relies on the hardness of PEP (or problems that reduce to PEP). Among the recent code-based signature proposals we consider, SPECK [3] is the only one that directly employs PEP through the PECK problem. LESS uses the more general LEP, and PERK uses the unrelated PKP.

This scope is built into the technique: Theorem 1 characterizes permutation-invariant bilinear forms, and permutation invariance is the symmetry of PEP but not of LEP or PKP. Extending the approach to LEP would require characterizing *monomial-invariant* bilinear forms, and Proposition 3 shows that the family $M = aI + bJ$ loses its invariance under non-uniform scaling. A characterization of monomial-invariant bilinear forms, and whether any of them yield hull reductions usable in an LEP setting, is left open.

6 Experimental Validation

Corollary 1 guarantees the dimension reduction for every nondegenerate $M = aI + bJ$ with $b \neq 0$, and the preprocessing reduces to a single rank query on a $k \times k$ matrix (Algorithm 1, Stage 1), which is $O(n^2)$ field operations in theory. What remains to be confirmed empirically is that (i) a single fixed choice (concretely $M = I + J$) suffices at SPECK parameters, removing any need to search over $(a, b) \in \mathbb{F}_q^2$, and (ii) the absolute wall-clock cost at $n = 252$ is small

Table 2. Empirical validation of Corollary 1. For each parameter set, we generated random self-dual codes and measured $\dim \mathrm{Hull}$ and $\dim \mathrm{Hull}_M$ with $M = I + J$. The "Red." column reports the observed dimension reduction (all equal to 1). The "Time" column is the average time to compute both hull ranks.

Parameters	Trials	$\dim \mathrm{Hull}$	$\dim \mathrm{Hull}_M$	Red.	Time (ms)
$n = 16$, $q = 5$	10	8	7	1	0.1
$n = 20$, $q = 7$	10	10	9	1	0.2
$n = 24$, $q = 11$	10	12	11	1	0.3
$n = 32$, $q = 127$	5	16	15	1	0.5
$n = 48$, $q = 127$	3	24	23	1	0.7
$n = 64$, $q = 127$	7	32	31	1	0.8
$n = 100$, $q = 127$	1	50	49	1	1.2
$n = 128$, $q = 127$	3	64	63	1	1.5
$n = 252$, $q = 127$	5	**126**	**125**	**1**	**2.0**
$n = 252$, $q = 8861$	3	**126**	**125**	**1**	**5.0**
Total	66 trials, 100% success				

enough that the factor-q speedup in the SSA main loop is not offset by hidden preprocessing. This section reports experiments on 66 self-dual codes, including SPECK instances, verifying both.

6.1 Setup

We implemented the generator-matrix construction $G = (I_k \mid A)$ with $AA^\top = -I_k$ in SageMath 10.0. For $q \equiv 1 \pmod 4$ (i.e., $q = 8861$), we take $A = \alpha I_k$ where $\alpha^2 = -1$ in $\mathbb{F}_q$. For $q \equiv 3 \pmod 4$ (i.e., $q = 127$), we use $k/2$ block-diagonal copies of $\left(\begin{smallmatrix} a & b \\ -b & a \end{smallmatrix}\right)$ with $a^2 + b^2 = -1$. Each resulting generator matrix is then randomized by left-multiplication with a random $U \in GL_k(\mathbb{F}_q)$ and coordinate permutation. For each sampled code C, we compute $\dim \mathrm{Hull}(C) = k - \mathrm{rank}(GG^\top)$ and $\dim \mathrm{Hull}_M(C) = k - \mathrm{rank}(GMG^\top)$ via Sage's rank routine.

6.2 Results

Table 2 reports the measured hull dimensions and preprocessing times across several parameter sets. The bilinear hull dimension is one less than the standard hull dimension in all 66 trials, and $v = G\mathbf{1}^\top$ is non-zero throughout, consistent with Corollary 1. The preprocessing time stays in the millisecond range even at SPECK parameters.

6.3 Discussion

Preprocessing cost. Three observations on the preprocessing:

1. *No search over (a, b) is required.* The fixed choice $M = I + J$ (i.e., $a = b = 1$) achieves the reduction guaranteed by Corollary 1; other (a, b) choices give the same $\dim \mathrm{Hull}_M$ for self-dual codes.
2. *The computation is a single rank query.* Since $GG^\top = 0$ for self-dual codes, $GMG^\top = bvv^\top$ has rank 1, and $\dim \mathrm{Hull}_M$ reduces to a single rank query on a $k \times k$ matrix, costing $O(k^2) = O(n^2)$ field operations.
3. *Wall-clock time is small.* At SPECK parameters ($n = 252$), preprocessing completes in 2–5 ms on a standard laptop, independent of q.

Thus the factor-q speedup in the SSA main loop is not offset by preprocessing: the $O(n^3 + n^2 q^{n/2-1} \log n)$ complexity of Theorem 3 holds end-to-end.

Scope of the experiments. We measure the preprocessing phase only. Running the full SSA on self-dual codes of dimension $n/2 = 126$ is out of reach: even with the factor-q improvement, the cost $q^{125} \gg 2^{1000}$ is beyond any hardware. The experiments therefore target the step that enables the factor-q speedup, namely the bilinear hull construction, rather than the subsequent SSA enumeration.

Consistency with theory. All 66 trials exhibit the predicted $\dim \mathrm{Hull}_M = \dim \mathrm{Hull} - 1$, consistent with Corollary 1: $n \not\equiv 0 \pmod{p}$ holds for both SPECK regimes ($\gcd(252, 127) = 1$ and $\gcd(252, 8861) = 1$). The experiments thus serve as a sanity check on our theoretical framework at SPECK parameters while simultaneously confirming the practicality of the preprocessing step. The core SageMath implementation is reproduced in Section A.

7 Conclusion

We studied, mathematically, how much the cost of the Support Splitting Algorithm can be lowered by replacing the standard inner product with an arbitrary bilinear form. Two structural results give a complete answer: Hull_M is permutation-invariant (and hence usable as an SSA invariant) if and only if $M = aI + bJ$ (Theorem 1), and every such M changes the hull dimension by at most one (Theorem 2, Corollary 1). The combined statement is a *tight upper bound on the bilinear-form route*: no bilinear generalization of the inner product can lower the SSA cost by more than a factor of q. Going beyond this would require invariants outside the bilinear framework, such as higher-order tensor invariants or non-bilinear algebraic structures.

We then surveyed the applicability to current code-based signature candidates. Among them, only PEP-based SPECK admits the factor-q SSA cost reduction on every self-dual instance ($\log_2 q \approx 13$ at the recommended $q = 8861$); LEP-based LESS does not extend (Proposition 3: $aI + bJ$ loses invariance under monomial transformations), and PKP-based PERK does not use hulls at all. A SageMath implementation of the bilinear-hull preprocessing confirms that the framework is practically realizable.

Future work. Several directions remain open.

- *Beyond bilinear invariants.* Whether higher-order tensor invariants or non-bilinear algebraic structures can reduce the SSA-relevant hull dimension by more than one is open. A positive answer would step outside the limit established here and represent a genuinely new technique against PEP-based schemes.
- *Other equivalence problems.* Recent work by Couvreur and Levrat [9] shows that hull-type invariants also play a role in solving the Matrix Code Equivalence Problem underlying schemes such as MEDS and ALTEQ. Whether an analogue of our $aI + bJ$ characterization exists in the matrix/rank-metric setting, and whether it leads to a similar cost reduction, is open. Closely related is the hull-variation problem of Ho and Johnsen [10] on equivalent vector rank-metric codes.
- *Hybrid Low-q analysis.* In SPECK's Low q regime the PEP-to-PECK reduction fails because $N_{\mathrm{sol}} \gg 1$. A precise hybrid analysis combining the bilinear-hull SSA on PEP with PECK-constraint filtering is left open (Section 4.5).

Acknowledgments. The authors thank the anonymous reviewers of SAC 2026 for their helpful comments. Y. A. was supported in part by JSPS KAKENHI Grant Number 24K20771. K. I. was supported in part by JSPS KAKENHI Grant Number 25K17290. T. M. was supported in part by JSPS KAKENHI Grant Number 25K24402.

A SageMath Implementation of the Validation Experiment

For reproducibility, below is the core of our SageMath implementation used for the experiments in Section 6. It validates Corollary 1 at SPECK parameters ($n = 252$, $q \in \{127, 8861\}$) in a few lines of linear algebra.

```
from sage.matrix.special import block_diagonal_matrix, ones_matrix

def construct_selfdual(n, q):
    """Build a self-dual [n, n/2]_q code G = (I_k | A) with A A^T = -I_k.

    q == 1 (mod 4): A = alpha * I_k, with alpha^2 = -1 in F_q.
    q == 3 (mod 4): block-diagonal of [[a, b], [-b, a]] with a^2 + b^2 = -1.
    """
    F = GF(q)
    k = n // 2
    I_k = identity_matrix(F, k)
    if q % 4 == 1:
```

```
        A = F(-1).sqrt() * I_k
    else:
        assert k % 2 == 0
        for a in F:
            rhs = F(-1) - a * a
            if rhs.is_square():
                b = rhs.sqrt()
                break
        block = matrix(F, [[a, b], [-b, a]])
        A = block_diagonal_matrix([block] * (k // 2))
    return I_k.augment(A)

def dim_hull(G, M=None):
    """dim Hull_M(C) = k - rank(G M G^T); M = None uses the standard
    inner product."""
    F = G.base_ring()
    if M is None:
        M = identity_matrix(F, G.ncols())
    return G.nrows() - (G * M * G.transpose()).rank()

# Validate Corollary 1 at SPECK parameters
for q in [127, 8861]:
    n = 252
    G = construct_selfdual(n, q)
    M = identity_matrix(GF(q), n) + ones_matrix(GF(q), n)  # M = I + J
    h_std = dim_hull(G)       # expect n/2 = 126
    h_bil = dim_hull(G, M)    # expect n/2 - 1 = 125
    print(f"q={q}: dim Hull = {h_std}, dim Hull_M = {h_bil}")
```

Sample output:

```
q=127:  dim Hull = 126, dim Hull_M = 125
q=8861: dim Hull = 126, dim Hull_M = 125
```

Both executions complete in under 5 ms of linear algebra.

References

1. Aaraj, N., et al.: PERK. In: NIST Post-Quantum Cryptography Round 2 Additional Signatures (2024)
2. Babai, L.: Graph isomorphism in quasipolynomial time. In: Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC) 2016. pp. 684–697 (2016)
3. Baldi, M., Battagliola, M., Mechri, R.E., Santini, P., Schiavoni, R., Zuane, D.D.: SPECK: Signatures from permutation equivalence of codes and kernels. Cryptology ePrint Archive, Paper 2025/923 (2025), https://eprint.iacr.org/2025/923
4. Baldi, M., Mechri, R.E., Santini, P., Schiavoni, R.: On the representation of self-orthogonal codes and applications to cryptography. Cryptology ePrint Archive, Paper 2025/2279 (2025), https://eprint.iacr.org/2025/2279

5. Bardet, M., Otmani, A., Saeed-Taha, M.: Permutation code equivalence is not harder than graph isomorphism when hulls are trivial. In: IEEE International Symposium on Information Theory (ISIT) 2019. pp. 2464–2468 (2019)
6. Barenghi, A., Biasse, J.F., Persichetti, E., Santini, P.: On the computational hardness of the code equivalence problem in cryptography. Advances in Mathematics of Communications **17**(1), 23–55 (2023)
7. Beullens, W.: Not enough LESS: An improved algorithm for solving code equivalence problems over $\mathbb{F}_q$. In: Selected Areas in Cryptography (SAC) 2020. LNCS, vol. 12804, pp. 387–403. Springer (2021)
8. Biasse, J.F., Micheli, G., Persichetti, E., Santini, P.: LESS is more: Code-based signatures without syndromes. In: Progress in Cryptology – AFRICACRYPT 2020. LNCS, vol. 12174, pp. 45–65. Springer (2020)
9. Couvreur, A., Levrat, C.: Highway to hull: An algorithm for solving the general matrix code equivalence problem. Cryptology ePrint Archive, Paper 2025/596 (2025), `https://eprint.iacr.org/2025/596`
10. Ho, D., Johnsen, T.: On the hull-variation problem of equivalent vector rank-metric codes. arXiv:2505.08506 (2025), `https://arxiv.org/abs/2505.08506`
11. Jitman, S., Sangwisut, E.: The average dimension of the Hermitian hull of constacyclic codes over finite fields of square order. Advances in Mathematics of Communications **12**(3), 451–463 (2018)
12. Leon, J.S.: Computing automorphism groups of error-correcting codes. IEEE Trans. Inf. Theory **28**(3), 496–511 (1982)
13. Loidreau, P., Sendrier, N.: Weak keys in the McEliece public-key cryptosystem. IEEE Trans. Inf. Theory **47**(3), 1207–1211 (2001)
14. Panny, L.: On breaking McEliece keys using brute force. In: Post-Quantum Cryptography (PQCrypto) 2026. LNCS, vol. 16492, pp. 275–308. Springer (2026), cryptology ePrint Archive, Paper 2025/632
15. Persichetti, E., Santini, P.: A new formulation of the linear equivalence problem and shorter LESS signatures. In: ASIACRYPT 2023, Part VII. LNCS, vol. 14444, pp. 351–378. Springer (2023)
16. Petrank, E., Roth, R.M.: Is code equivalence easy to decide? IEEE Trans. Inf. Theory **43**(5), 1602–1604 (1997)
17. Santini, P., Baldi, M., Chiaraluce, F.: Computational hardness of the permuted kernel and subcode equivalence problems. IEEE Trans. Inf. Theory **70**(3), 2254–2270 (2024). `https://doi.org/10.1109/TIT.2023.3323068`
18. Sendrier, N.: On the dimension of the hull. SIAM J. Discrete Math. **10**(2), 282–293 (1997)
19. Sendrier, N.: Finding the permutation between equivalent linear codes: The support splitting algorithm. IEEE Trans. Inf. Theory **46**(4), 1193–1203 (2000). `https://doi.org/10.1109/18.850662`
20. Sendrier, N., Simos, D.E.: The hardness of code equivalence over $\mathbb{F}_q$ and its application to code-based cryptography. In: Post-Quantum Cryptography (PQCrypto) 2013. LNCS, vol. 7932, pp. 203–216. Springer (2013)
21. Skersys, G.: The average dimension of the hull of cyclic codes. Discrete Applied Mathematics **128**(1), 275–292 (2003)