On products of linear error correcting codes

Mirandola, D.

Chapter 1

A Survey on Code Products

1.1 Codes and Code Products

In this thesis we study products of linear error correcting codes. We show three main results on such products and discuss applications to cryptography. Our methods are typically algebraic-combinatorial in nature, though sometimes probabilistic techniques will be involved. In this survey chapter we introduce codes and code products, and motivate our interest by showing how code products have appeared and are relevant in several topics. Finally, we will conclude this chapter with an overview of our results and a discussion putting their significance into perspective.

The following scenario, outlined by MacWilliams and Sloane in their handbook [48], may appear slightly old fashioned, but still helps introducing error correcting codes, as a mean to correct the errors introduced by some noisy communication channel.

> Suppose there is a telegraph wire from Boston to New York down which 0's and 1's can be sent. Usually when a 0 is sent it is received as a 0, but occasionally a 0 will be received as a 1, or a 1 as a 0. Let's say that [...] for each symbol there is a probability $p = 1/100$ that the channel will make a mistake.

Modern settings in which error correcting codes are used are for instance deep space communications, broadcasting and mass storage. These share a common

feature: retransmission of data is impossible, due to economic or practical constraints. As an example, suppose that, ten years ago, we recorded our favourite song on a disc, and now we want to listen to that song again. If we had done it "naively", that is if we saved one bit of the song (whatever it means) as one bit of information in the disc (whatever it means), then we would have no way to recover corrupted bits. Note that in this case retransmission, i.e. asking our ten-years-ago self to record the disc again, is not an option.

Roughly speaking, an error correcting code is given by a pair of functions `Enc` and `Dec`, standing for encode and decode respectively, with the following property: if a message m is encoded as $x = \texttt{Enc}(m)$, and x is turned into $\tilde{x}$ by a "small" error e, then $\tilde{x}$ is correctly decoded as $\texttt{Dec}(\tilde{x}) = m$. This situation is represented in Figure 1.1. Of course it shall be ensured that $\texttt{Dec}(\texttt{Enc}(m)) = m$, i.e. decoding always works properly if no corruption occurred.

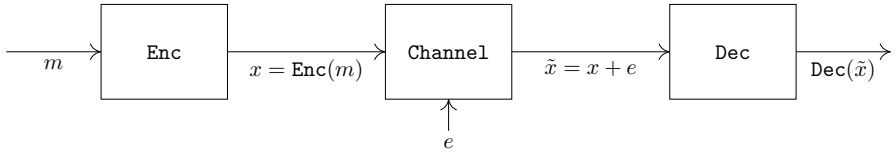

Figure 1.1

Refraining again from a proper mathematical formalization, we give an intuition of how encoding and decoding are possible. The encoding function `Enc` embeds a set $\mathcal{M}$ of allowed messages into a larger set $\mathcal{E}$ which contains, beside the set $\texttt{Enc}(\mathcal{M})$ of all meaningful encodings, all their corrupted variants. In addition, $\mathcal{E}$ is endowed with a metric structure, i.e. a notion of distance between any two elements of $\mathcal{E}$ is defined. In particular, this allows us to quantify an error, by measuring the distance between an encoding x and its corrupted version $\tilde{x}$. Now, assume that `Enc` maps the elements of $\mathcal{M}$ into elements of $\mathcal{E}$ which are sufficiently far apart from each other, with respect to this notion of distance. As above, assume that a message m is encoded as $x = \texttt{Enc}(m)$ and turned into $\tilde{x}$ by some error. If the error is sufficiently small, then x will be uniquely identified as the closest-to-$\tilde{x}$ element of $\texttt{Enc}(\mathcal{M})$, and the original message computed as $m = \texttt{Enc}^{-1}(x)$. Figure 1.2 represents this situation.

We are finally ready to formalize this setting. Prominent notions are those of linear code and Hamming distance, which model the copy of $\mathcal{M}$ in $\mathcal{E}$ containing all meaningful encodings and the metric structure of the ambient space $\mathcal{E}$. We will not be concerned with more general families of codes or with different metrics in this work. A wider introduction to the theory of linear error correcting codes is given in Section 2.5. Among the standard references on the topic we cite [37, 48, 71].

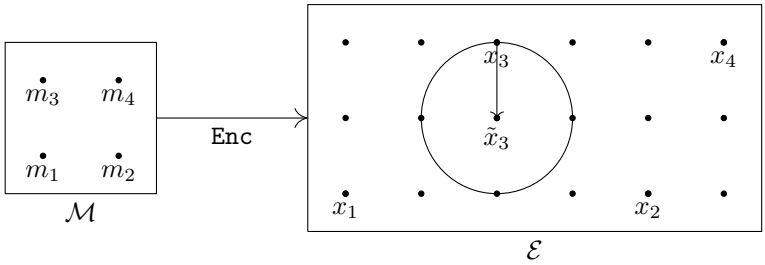

Figure 1.2: The message space is $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$, its image is $\texttt{Enc}(\mathcal{M}) = \{x_1, x_2, x_3, x_4\} \subseteq \mathcal{E}$. The message m_3 is encoded as $x_3 = \texttt{Enc}(m_3)$ and transmitted, then turned into $\tilde{x}_3$ by some error. As x_3 is the closest element of $\texttt{Enc}(\mathcal{M})$, $\texttt{Dec}(\tilde{x}_3) = \texttt{Enc}^{-1}(x_3) = m_3$ is decoded correctly.

Let $\mathbb{F}$ be a finite field and let q denote its size[1]. Let $k \leq n$ be two positive integers. The encoding function $\texttt{Enc}$ maps messages from the $\mathbb{F}$-vector space $\mathcal{M} := \mathbb{F}^k$ into elements of the $\mathbb{F}$-vector space $\mathcal{E} := \mathbb{F}^n$. It is required to be injective, so that any encoded message can be unambiguously recovered. The image $C := \texttt{Enc}(\mathcal{M}) \subseteq \mathcal{E}$ is called a *code*, and it is *linear* if the encoding function is $\mathbb{F}$-linear. The elements of a code are called *codewords*. If this is the case, we can associate a $k \times n$ matrix G, a *generator matrix* of C, to the linear map $\texttt{Enc}$ so that $\texttt{Enc}(m) = mG$ and $C = \{mG : m \in \mathbb{F}^k\} \cong \mathbb{F}^k$. Here we write vectors in row form as it is customary in coding theory.

The *(Hamming) distance* between two vectors

$$x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \mathbb{F}^n$$

is

$$d(x, y) := |\{i : x_i \neq y_i\}|,$$

that is the number of positions in which x and y differ[2]. The *weight* of a vector is its distance from the zero vector, i.e. the number of positions in which it has a non-zero entry. The *minimum distance* of the code C is

$$d_{\min}(C) := \min\{d(x, y) : x, y \in C, x \neq y\} = \min\{\mathrm{wt}(x) : x \in C, x \neq 0\},$$

that is the minimal distance between any two distinct codewords, or equivalently the minimal weight of any non-zero codeword.

The decoding function $\texttt{Dec}$, for all $x \in \mathbb{F}^n$, is defined as follows: if there exists a unique $y \in C$ which minimizes $d(x, y)$ then $\texttt{Dec}(x) := \texttt{Enc}^{-1}(y)$;

[1] It is a well-known fact that q is a prime power.

[2] The Hamming distance between two vectors is always a non-negative integer and is indeed a distance in the usual mathematical sense: for any $x, y, z \in \mathbb{F}^n$ it holds that (i) $d(x, y) \geq 0$, and $d(x, y) = 0$ if and only if $x = y$, (ii) $d(x, y) = d(y, x)$, (iii) $d(x, y) \leq d(x, z) + d(z, y)$.

otherwise $\texttt{Dec}(x) := \perp$, an abort symbol. Observe that, even though such a function is well defined[3], it may be practically unfeasible to compute it by exhaustive search as the definition seems to require. Efficient, specific decoding algorithms are used instead in all applications: for instance, if a code has a t-error correcting pair (see the next section) then it has a t-error correcting algorithm with complexity $O(n^3)$. The minimum distance quantifies the error tolerance of a code. Errors are modeled as vectors which are added to the message: a vector $x \in \mathbb{F}^n$, corrupted by $e \in \mathbb{F}^n$, becomes $\tilde{x} = x + e$, and in this case we say that $\mathrm{wt}(e)$ errors occurred. It is easy to see that, for all $m \in \mathbb{F}^k$ and $e \in \mathbb{F}^n$ with $\mathrm{wt}(e) < d_{\min}(C)/2$, we have

$$\texttt{Dec}(\texttt{Enc}(m) + e) = m,$$

i.e. a code can tolerate errors of weight up to half of its minimum distance.

We continue with a remark about the relevant parameters of a code, namely length, dimension (as an $\mathbb{F}$-vector space) and minimum distance. For fixed length, it is of course desirable for dimension and minimum distance to be as large as possible, as these measure the size of the messages that we can encode and the amount of errors that we can tolerate. The trade-off between them is quantified by several classical bounds. Among them, the *Singleton Bound* claims that, for a code C of length n, it holds that

$$\dim C + d_{\min}(C) \leq n + 1.$$

If C attains this bound, i.e. if

$$\dim C + d_{\min}(C) = n + 1,$$

then it is said to be *maximum distance separable (MDS)*.

In order to define the product of two codes, we need to define some additional structure on the ambient space. Observe that the n-fold cartesian product $\mathbb{F}^n$ has a natural structure of $\mathbb{F}$-algebra, with multiplication induced by componentwise application of multiplication in $\mathbb{F}$, i.e. for all $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \mathbb{F}^n$ we define

$$xy := (x_1 y_1, \ldots, x_n y_n).$$

We also define $x^2 := xx$ and higher powers in the obvious way.

Given two codes $C, D \subseteq \mathbb{F}^n$, their *product* is the $\mathbb{F}$-linear span of the set of all products xy with $x \in C$ and $y \in D$,

$$CD := \langle xy : x \in C, y \in D \rangle.$$

[3]As $\texttt{Enc}$ is injective, $\texttt{Enc}^{-1}(y)$ is well defined for all $y \in C$.

Observe that the set of all products is not necessarily additively closed, so it is strictly contained in the code product in general. We also define the *square* $C^2 := CC$ of a code, and higher powers inductively. We are using the same notation for set cartesian product and code componentwise product, but the context will always help clarifying this ambiguity.

The product of two codes $C, D \subseteq \mathbb{F}^n$, sometimes called the *Schur product*, has usually been denoted by $C * D$, but we shall drop the star symbol to lighten notation. Products of codes turn up in a variety of situations, such as algebraic error correction, secret sharing and multiparty computation, algebraic complexity theory, additive combinatorics, and lately cryptanalysis. This survey will briefly encompass all these topics. A number of efforts have gone into describing the code-theoretic structure of code products, see [28, Chapter 12] and [65] for an extensive review of the current state of the art. In particular, [65] collects several technical results which will be cited explicitly and used in this thesis.

We can immediately state a trivial upper bound for the product dimension. For any pair of codes C, D it holds that

$$\dim CD \leq \dim C \dim D \quad \text{and} \quad \dim C^2 \leq \frac{\dim C(\dim C + 1)}{2}.$$

To see this, observe that if $\{x_1, \ldots, x_k\}$ and $\{y_1, \ldots, y_\ell\}$, where $k := \dim C$ and $\ell := \dim D$, are $\mathbb{F}$-bases of C and D respectively then the elements $x_i y_j$ with $1 \leq i \leq k, 1 \leq j \leq \ell$ generate CD and the elements $x_i x_j$ with $1 \leq i \leq j \leq k$ generate C^2. In fact it holds that these bounds are achieved by most codes: roughly speaking, code products typically fill the whole space. This is shown in Chapter 3, which is based on [13], for the second inequality, while for the first inequality the reader is referred to [64].

We conclude this section with an example. The codes we are going to describe not only have a nice mathematical structure, but are also widely used in practical applications. Fix $k \leq n \leq q$ and n pairwise distinct elements $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$. Let $\mathbb{F}[X]_{<k}$ denote the vector space of all polynomials in the indeterminate X, with coefficients in $\mathbb{F}$, and degree less than k. The image of the evaluation map

$$\begin{array}{ccc} \mathbb{F}[X]_{<k} & \hookrightarrow & \mathbb{F}^n \\ f & \mapsto & (f(\alpha_1), \ldots, f(\alpha_n)) \end{array}$$

is a linear space, called a *Reed-Solomon code*. This map is injective because any polynomial of degree at most $k-1$ is uniquely determined by any k distinct evaluations, hence the code has dimension k. Moreover, a polynomial of degree at most $k-1$ has at most $k-1$ zeros, hence any codeword has weight at least

$n-k+1$. It follows that the code has minimum distance at least $n-k+1$, hence it is an MDS code.

The image of the standard basis of $\mathbb{F}[X]_{<k}$ is a basis of the code, and gives a generator matrix in Vandermonde form, namely

$$\begin{pmatrix} 1 & \cdots & 1 \\ \alpha_1 & \cdots & \alpha_n \\ \vdots & & \vdots \\ \alpha_1^{k-1} & \cdots & \alpha_n^{k-1} \end{pmatrix}.$$

Now let C and D be Reed-Solomon codes of length n with the same evaluation points $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$. It is easy to see that also CD is a Reed-Solomon code with the same evaluation points and that

$$\dim CD = \dim C + \dim D - 1,$$

provided that this quantity is smaller than n. Indeed, if C and D are the images of $\mathbb{F}[X]_{<k}$ and $\mathbb{F}[X]_{<\ell}$ respectively, where $k := \dim C$ and $\ell := \dim D$, then CD is the image of $\mathbb{F}[X]_{<k+\ell-1}$, because $\mathbb{F}[X]_{<k+\ell-1}$ is spanned by the polynomials of the form fg with $f \in \mathbb{F}[X]_{<k}, g \in \mathbb{F}[X]_{<\ell}$. Observe that in this case the dimension is significantly smaller than the general upper bound obtained above[4].

The rest of this survey is dedicated to motivating our systematic code-theoretic study of code products, by showing a number of different contexts in which questions related to their possible parameters arise. An outline of the structure of this thesis concludes the chapter.

1.2 Error Locating Pairs

Possibly one of the earliest appearances of code products goes back to [57, 58, 59, 43] where it is relevant to the notion of error locating pairs used for algebraic decoding. On a historical note, we mention earlier appearances of code products in work on a proof of the Roos bound for cyclic codes [70] and on secure multiparty computation [6, 18].

Throughout this section, let t denote a positive integer. According to [57, 58], a *t-error locating pair* for a code $C \subseteq \mathbb{F}^n$ is a pair of codes $A, B \subseteq \mathbb{F}^n$ satisfying

(i) $AB \subseteq C^\perp$,

[4] We are comparing $\dim C + \dim D - 1$ with $\dim C \dim D$.

(ii) $\dim A > t$,

(iii) $d_{\min}(B^{\perp}) > t$.

Here the symbol "$\perp$" denotes the dual with respect to the standard inner product in $\mathbb{F}^n$. Observe that the product of A and B appears in the first property. If in addition it holds that

(iv) $d_{\min}(A) + d_{\min}(C) > n$

then the pair is said to be *t-error correcting*. In [59, 60] this definition was extended by allowing A and B to be defined over a finite extension of $\mathbb{F}$.

As an example, consider a pair of Reed-Solomon codes A and B with the same sequence of evaluation points. Assume that $\dim A = t + 1$ and $\dim B = t$. Then (A, B) is a t-error correcting pair for $C := (AB)^{\perp}$. Other constructions of error correcting pairs can be found in [58, 60] for algebraic-geometric codes and in [31] for cyclic codes.

These objects are relevant to the decoding problem. Suppose that the sum $\tilde{x} = x + e$ of a codeword $x \in C$ and of an error vector $e \in \mathbb{F}^n$ of weight t is known. Is it possible to correct the t errors in $\tilde{x}$, i.e. recover x, efficiently? The existence of error correcting pairs allows one to answer positively [57, 58]: given a t-error correcting pair, it is possible to build a t-error correcting algorithm with complexity $O(n^3)$, where n denotes the length of the code.

We show how this works in practice for a Reed-Solomon code C. Recall that in this case a codeword is of the form $x = (f(\alpha_1), \ldots, f(\alpha_n))$, where $f \in \mathbb{F}[X]_{<k}$ is a polynomial of degree less than k and $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ are pairwise distinct. Suppose that the vector

$$\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_n) \in \mathbb{F}^n$$

is received, and that $\mathrm{wt}(\tilde{x} - x) \leq t$, i.e. at most t errors occurred. Our purpose is to recover the original codeword x, or equivalently the error vector $e := \tilde{x} - x$. The key observation is the following: a polynomial $\ell \in \mathbb{F}[X]$ of degree t which is zero at all error positions, i.e. $\ell(\alpha_i) = 0$ for all i such that $\tilde{x}_i - f(\alpha_i) \neq 0$, satisfies

$$f(\alpha_i)\ell(\alpha_i) = \tilde{x}_i \ell(\alpha_i)$$

for all $i = 1, \ldots, n$. This is a system of n equations which is quadratic in the coefficients of the polynomials f and ℓ. Let A denote the Reed-Solomon code corresponding to the polynomial space $\mathbb{F}[X]_{<t+1}$ with evaluation points $\alpha, \ldots, \alpha_n$, so that $(\ell(\alpha_1), \ldots, \ell(\alpha_n)) \in A$. Observe that at the left-hand side we have the entries of the vector

$$(f(\alpha_1)\ell(\alpha_1), \ldots, f(\alpha_n)\ell(\alpha_n)),$$

which belongs to the product code CA. We can transform this quadratic system into a linear system by replacing the left-hand side with a polynomial $g \in \mathbb{F}[X]$ of degree at most $k + t$ which needs to satisfy

$$g(\alpha_i) = \tilde{x}_i \ell(\alpha_i)$$

for all $i = 1, \ldots, n$. Now the unknowns are the $k + 2t$ coefficients of the polynomials g and ℓ. Finally, if a solution of the form $g = f\ell$ is obtained, then $f = g/\ell$ can be recovered.

The codes C and A in our example correspond to C and A in the definition of an error correcting pair. The code B in the definition corresponds to the dual of the product code CA in our example, fulfilling the first requirement of the definition. The other conditions ensure that the quadratic system above has a solution, that such a solution is of the desired form and, finally, that it is unique.

1.3 Secret Sharing and Secure Multiparty Computation

"Products" and "squares" of codes are the primary focus of work on arithmetic secret sharing [19, 11, 15, 16] and its application to secure multi-party computation [27]. In this thesis we will not be concerned with notions of secret sharing without arithmetic properties, and the interested reader is referred to [5, 56]. Secret sharing has as main motivation and application secure multiparty computation (MPC). Secure multiparty computation studies the problem of evaluating a function on inputs submitted by several players, while guaranteeing privacy and correctness even in presence of dishonest players, who may try to acquire more information than they are supposed to, possibly deviating from the protocol.

Secret sharing deals with the problem of protecting a *secret* by distributing *shares* among a number of *players*, in a way so that only some privileged player coalitions are *accepted*, i.e. can recover the secret by putting together their shares, while other player coalitions are *rejected*, i.e. any possible secret is equally likely to them. An algebraic structure which implements this functionality is called a *secret sharing scheme*. The family of all accepted set and the family of all rejected set are called the *access structure* and the *adversary structure* respectively. A scheme has *t-privacy* if the adversary structure contains any set of (at most) t players, and has *r-reconstruction* if the access structure contains any set of (at least) r players. Here t and r are positive integers with $1 \leq t < r \leq n$, where n denotes the number of players. A scheme with $(r-1)$-privacy and r-reconstruction is called *r-threshold*.

To share a secret $s \in \mathbb{F}$ among n players using a linear code $C \subseteq \mathbb{F}^{n+1}$, one standardly chooses a random codeword whose 0-th coordinate equals s and define the i-th share to be the i-th coordinate[5] [50]. Then the privacy and reconstruction parameters of the scheme can be estimated from the parameters of the code: precisely, the scheme has $(d_{\min}(C^{\perp}) - 2)$-privacy and $(n - d_{\min}(C) + 2)$-reconstruction[6]. Analogously, to share a secret vector $s \in \mathbb{F}^k$ among n players using a linear code $C \subseteq \mathbb{F}^{n+k}$, one standardly chooses a random codeword with some fixed k-tuple of coordinates equal to s and distributes the other coordinates as shares. Again, privacy and reconstruction of the scheme can be estimated using the minimum distance of the dual and of the code itself respectively.

When two secrets s and s' are shared in this way, summing coordinatewise the share vectors gives naturally a share vector of the coordinatewise sum $s + s'$ of the secrets[7]. When one considers the product of the share vectors, one obtains a share of the product ss', but for a different secret sharing scheme, namely that associated to the product code C^2. We say that a secret sharing scheme is *arithmetic* if it supports multiplication, i.e. if the product of two secrets can be reconstructed from the product of the share vectors. To prevent a common misunderstanding, we highlight here that, in practical applications, the product reconstruction property is not used in the straightforward way, i.e. to recover the secret product given the share products. Instead, it allows to reduce a secure multiplication to a secure computation of a linear functional.

If C is a Reed-Solomon code, the above construction defines the well-known Shamir scheme [67]. Let $\alpha_1, \ldots, \alpha_n$ be non-zero, pairwise distinct elements of $\mathbb{F}$. To share a secret $s \in \mathbb{F}$, one picks uniformly at random a polynomial f of degree less than a fixed parameter k, under the constraint that $f(0) = s$, and defines the i-th share to be $f(\alpha_i)$. It turns out that the scheme has $(k-1)$-privacy and k-reconstruction, hence in particular it is k-threshold. Assuming that $2k - 1 \leq n$, we have that the code square C^2 is also a Reed-Solomon code, hence it defines a scheme with $(2k-1)$-reconstruction. It follows that Shamir's scheme is arithmetic in this case.

The above operational definition of secret sharing can be formalized in several equivalent ways. Among these we mention the notion of *codex*[8], introduced in [16] and extensively treated in [28]. For instance, using this definition, an $(n, t, 1, r)$-codex for $\mathbb{F}$ over $\mathbb{F}$ is a secret sharing scheme among n players with t-privacy and r-reconstruction, while an $(n, 1, 2, n)$-codex is an arithmetic secret sharing scheme among n players. An $(n, t, 2, n-t)$-codex is an arithmetic secret sharing scheme with t-privacy and $(n-t)$-product reconstruction, i.e.

[5] We index the coordinates of $\mathbb{F}^{n+1}$ with $\{0, 1, \ldots, n\}$.
[6] Here $C^{\perp}$ denotes the dual of C with respect to the standard inner product in $\mathbb{F}^{n+1}$.
[7] Because the code is linear.
[8] The plural of codex is *codices*.

the product of two secrets can be reconstructed from any set of $n - t$ products of shares. Such a secret sharing scheme is called *t-strongly multiplicative*. Roughly speaking, given an (n, t, d, r)-codex, n is the number of players, t is the privacy threshold, d is the multiplicative depth and r is the product reconstruction threshold. Moreover we can have codices for arbitrary $\mathbb{F}$-algebras, such as $\mathbb{F}^k$ or finite extension fields of $\mathbb{F}$, meaning that the secret lies in this algebra. The strength of this notion is that it encompasses all known relevant variations on arithmetic secret sharing, and notions from other fields, such as the one of bilinear multiplication algorithm introduced in Section 1.4, as well. In addition, in [28, Section 12.5.4] codices are used to present a variation on the decoding method based on error correcting pairs.

Since the parameters of a code are relevant to the associated secret sharing scheme, studying the parameters of C^2 becomes important. In order to be useful for strongly multiplicative secret sharing, a code needs to have a dual with good minimum distance (to control the privacy threshold) and a square with good minimum distance as well (to control the product reconstruction threshold). Hence interest is focused on families of linear codes $(C_i)_{i\in\mathbb{N}}$ of unbounded length, such that the families of the dual codes $(C_i^\perp)_{i\in\mathbb{N}}$ and of the squares $(C_i^2)_{i\in\mathbb{N}}$ are asymptotically good, i.e.

$$\limsup_{i\to\infty} \frac{d_{\min}(C_i^\perp)}{n_i} > 0, \quad \limsup_{i\to\infty} \frac{d_{\min}(C_i^2)}{n_i} > 0,$$

where, for all $i \in \mathbb{N}$, n_i denotes the length of C_i [11].

Such families were first constructed, over all finite fields of size $q \geq 49$ with q square, in [19] using techniques from algebraic geometry, namely asymptotically good towers of algebraic function fields. In [11] these families of codes were combined with a dedicated field descent technique to obtain arithmetic secret sharing schemes with good parameters over any field[9]. This work was subsequently extended in [15, 17], with the construction of asymptotically good families of codes over fields of size $q = 8, 9$ and $q \geq 16$, involving novel algebraic-geometric ideas such as torsion limits and Riemann-Roch systems of equations for function fields. We remark that no elementary construction of such families of codes is known so far.

Besides its original application, the result of [19] played a central role in the paper [40] on the "secure MPC in the head" paradigm: here secure MPC is used as an abstract primitive for efficient two-party cryptography[10]. Among other subsequent fundamental results, let us mention that asymptotically good codes whose dual and square are also asymptotically good are an essential in-

[9] The corresponding codes may be bad.

[10] For an extensive treatment of the interplay between secure multiparty computation, (arithmetic) secret sharing, codes and algebraic geometry, please consult [28].

gredient in the recent constructions of efficient unconditionally secure oblivious transfer protocols from noisy channels [38].

Bilinear complexity theory, briefly discussed in Section 1.4, is concerned with a similar problem, namely the construction of asymptotically good families of codes whose squares are also asymptotically good. As opposed to the case of secret sharing, in this setting no condition is imposed on the duals. Such families have been shown to exist for all finite fields in [63]. This construction carefully combines algebraic geometric codes that have asymptotically good higher powers, which can be constructed over large enough finite fields, with a field descent concatenation technique. Again, no elementary construction is known in this case.

Finally, recent work [1], inspired by [53], exploited combinatorial properties of codes and code products to prove that, among all t-strongly multiplicative secret sharing schemes on n players, only Shamir's scheme can achieve the optimal $t = (n-1)/3$.

1.4 Bilinear Multiplication Algorithms

Code products also appear in algebraic complexity theory [21]. There one wishes to express multiplication in some finite extension field $\mathbb{L}/\mathbb{F}$ through a bilinear algorithm involving a small number of multiplications in $\mathbb{F}$: given $x, y \in \mathbb{L}$, instead of computing their product directly, one wants to map them into $\mathbb{F}^n$ using a linear map σ, componentwise multiply $\sigma(x)$ and $\sigma(y)$, and then map their product back to $\mathbb{L}$ using another linear map ρ. The requirement is that

$$xy = \rho(\sigma(x)\sigma(y))$$

for all $x, y \in \mathbb{L}$, where the multiplication at the left-hand side is in $\mathbb{L}$ while the multiplication at the right-hand side is in $\mathbb{F}^n$. In other words, the following diagram has to be commutative.

$$\begin{array}{ccccc}
\mathbb{F}^n & & \mathbb{F}^n & & \mathbb{F}^n \\
\cup\!| & & \cup\!| & & \cup\!| \\
C & \times & C & \longrightarrow & C^2 \\
\sigma \uparrow & & \sigma \uparrow & & \downarrow \rho \\
\mathbb{L} & \times & \mathbb{L} & \longrightarrow & \mathbb{L}
\end{array}$$

To better highlight how this topic is related to code squares, we remark that the image C of $\mathbb{L}$ via σ is a linear subspace of $\mathbb{F}^n$, i.e. a code, and that the

image of C via the componentwise multiplication in $\mathbb{F}^n$ spans C^2.

The pair (σ, ρ) is called a *bilinear multiplication algorithm* for $\mathbb{L}$ over $\mathbb{F}$, and n is its *expansion*. The minimal among the expansions of all bilinear multiplication algorithms for $\mathbb{L}$ over $\mathbb{F}$ is called the *bilinear complexity* of $\mathbb{L}$ over $\mathbb{F}$. If a bilinear multiplication algorithm for $\mathbb{L}$ over $\mathbb{F}$ with expansion n exists, then we can reduce multiplication in $\mathbb{L}$ to n multiplications in $\mathbb{F}$ (and application of two linear maps).

As an example, textbook multiplication in $\mathbb{L}$, which consists of identifying elements of $\mathbb{L}$ with univariate polynomials with coefficients in $\mathbb{F}$ and multiplying them as such, is a bilinear multiplication algorithm with expansion $n = \binom{k+1}{2}$, where k denotes the degree of the field extension.

As anticipated in the previous section, this notion is encompassed by the codex definition: a bilinear multiplication algorithm for $\mathbb{L}$ over $\mathbb{F}$ is an $(n, 0, 2, n)$-codex for $\mathbb{L}$ over $\mathbb{F}$. Recall that an $(n, 0, 2, n)$-codex is a secret sharing scheme among n players with 0-privacy and n-product reconstruction. In order to obtain a bilinear multiplication algorithm from such a scheme, it suffices to define σ to be the map which assigns to a secret a set of valid shares, and ρ the map which reconstructs the product of two secrets from the products of the shares. In addition, we require that the dimension of $\sigma(\mathbb{L})$ as an $\mathbb{F}$-vector space equals the degree of $\mathbb{L}$ as an extension field of $\mathbb{F}$. This prevents redundancies in the bilinear multiplication algorithm.

Among the first results on the topic, we mention [74] and [46]. In [74] it is proved that any bilinear multiplication algorithm has expansion $n \geq 2k - 1$, where k denotes the degree of the field extension. In [46] it is proved that the bilinear complexity is a quasi-linear function of the extension degree k, i.e. it is bounded by $f(k)k$ where f satisfies

$$f(k) < \log \log \cdots \log k$$

for any number of applications of the logarithm function. For recent developments, we refer to [3, 14, 61, 17].

1.5 Additive Combinatorics

Additive combinatorics [69] investigates the additive structure of sets. Given an abelian group G and two non-empty subsets $A, B \subseteq G$, additive combinatorics studies, for instance, the size of the sum set

$$A + B := \{a + b : a \in A, b \in B\}$$

and the necessary conditions so that the sum set size is minimal. This problem is the object of the classical theorems of Kneser [42] and Vosper [72]. For background on and proofs of Kneser and Vosper's Theorems we refer to [69]. Kneser's Theorem implies in particular that if A, B are subsets of an abelian group such that

$$|A + B| < |A| + |B| - 1$$

then $A + B$ must be periodic, i.e. there exists a non-zero element g of the abelian group that stabilizes $A + B$ so that we have $A + B + g = A + B$. Vosper's Theorem is a characterization of pairs of subsets A, B of the integers modulo a prime p with the property that $|A + B| = |A| + |B| - 1$. It states that, excluding some degenerate cases, A, B must be arithmetic progressions with the same difference.

The purpose of some recent works [36, 2, 4, 53] is to translate questions from classical additive combinatorics to different contexts. As an example, one can take a field extension $\mathbb{L}/\mathbb{K}$ instead of an abelian group as ambient space. In this context, we can consider two $\mathbb{K}$-vector spaces S, T contained in $\mathbb{L}$ and study the dimension of the product vector space

$$ST := \langle st : s \in S, t \in T \rangle,$$

where the product is the field multiplication in $\mathbb{L}$ and the brackets $\langle \cdot \rangle$ mean that the linear span is taken. It was proved in [36] that an analogue of Kneser's Theorem carries over to this case[11].

A subsequent step is to translate additive combinatorics into the context of coding theory: consider $\mathbb{F}^n$, where $\mathbb{F}$ is finite field, as ambient space, and let $C, D \subseteq \mathbb{F}^n$ be two $\mathbb{F}$-vector spaces, i.e. two codes. In this setting, the natural counterpart of the sum set size is the dimension of the code product CD. An even more general context is considered in [4], where $\mathbb{F}^n$ is replaced by an arbitrary algebra over the base field.

1.6 Cryptanalysis of McEliece Cryptosystem

As a last motivation, there has been some recent use of code squares in the cryptanalysis of variants of the McEliece cryptosystem. McEliece cryptosystem [52] is a code-based public-key cryptosystem which relies on the hardness of the general decoding problem [8].

Let C be a code, with encoding and decoding algorithm `Enc` and `Dec`, and assume that `Dec` can correct efficiently t errors. For instance, one may think that C admits a t-error correcting pair. Then a secret message m can be

[11] Under the additional assumption that the field extension is separable.

encrypted as $c := \texttt{Enc}(m) + e$, where e is a random vector of weight t. Due to the error correcting property of the algorithm, it is possible to recover the original message as

$$m = \texttt{Dec}(c) = \texttt{Dec}(\texttt{Enc}(m) + e).$$

An external adversary (who does not know Dec, or in our example a t-error correcting pair for C), in order to recover m, is required to solve the general decoding problem, which is known to be hard.

Concretely, the private key consists of C and Dec, while the public key is a generator matrix G of C together with the decoding capability t of Dec. The matrix G is "scrambled" in a way so that the original structure of the code is hidden[12], and consequently the efficient decoding algorithm as well,. To build this cryptosystem, Goppa codes [48, Chapter 12] are standardly used. One immediately notices that the public key, being a matrix, is huge: this is the main disadvantage of this cryptosystem.

The main advantage is the reliance on the general decoding problem, which makes this cryptosystem resistant even in a post-quantum scenario. On the other hand, recent attacks aim to recover the "hidden" structure of the code from the "scrambled" matrix, hence the efficient decoding algorithm, rather than the original message directly via general decoding algorithms. The idea exploited in [34, 23, 25, 26] is that Goppa codes have a square that has a substantially smaller dimension than typical random linear codes: this allows to build a distinguisher which can be used to attack the cryptosystem.

As an example, we quickly sketch how code squares were used in [24] to attack Wieschebrink's encryption scheme [73]. To give a bit of context, we recall that McEliece cryptosystem based on Reed-Solomon codes, as proposed in [55], was proved to be insecure in [68]: here it was shown that, in the case of a Reed-Solomon code, a generator matrix in standard form can be recovered efficiently from any scrambled one. To fix this, Wieschebrink [73] proposed to insert in the generator matrix some random columns: this suffices to make the algorithm [68] fail, while preserving the decryption capability of the code. This variant was broken in [24], using arguments based on code squares. The idea that is exploited is that the dimension of the square of a Reed-Solomon code C is

$$\dim C^2 = 2 \dim C - 1,$$

while in the general, random case the square of a code tends to fill the full space[13]. Let C be a code obtained by inserting in a Reed-Solomon code some random columns. Let i be a coordinate and let $C_{\overline{i}}$ be the code obtained

[12] To obtain such a matrix, one can take any generator matrix G' and define $G := HG'P$ where H is invertible and P is a permutation matrix.

[13] This is formalized and proved in Chapter 3, which is based on [13].

by puncturing C at i, i.e. the code obtained from C by removing the i-th coordinate of all its codewords. Now compare the dimension of the squares of C and $C_{\bar{i}}$: if the square dimension decreased after puncturing, i.e. if

$$\dim C_{\bar{i}}^2 < \dim C^2,$$

then the i coordinate corresponds to a random column. Iterating this argument, one can remove all random columns, and finally be able to apply [68] to recover the original Reed-Solomon code.

1.7 Outline of the Thesis

The main body of this thesis consists of three chapters, dedicated to the following three different published works.

[13] I. Cascudo, R. Cramer, D. Mirandola, and G. Zémor. Squares of Random Linear Codes. *IEEE Transactions on Information Theory*, 61(3):1159–1173, March 2015.

[53] D. Mirandola and G. Zémor. Critical Pairs for the Product Singleton Bound. *IEEE Transactions on Information Theory*, 61(9):4928–4937, Sept. 2015.

[12] I. Cascudo, R. Cramer, D. Mirandola, C. Padró, and C. Xing. On secret sharing with nonlinear product reconstruction. *SIAM Journal on Discrete Mathematics*, 29(2):1114–1131, 2015.

This is preceded by a preliminary chapter where all the mathematical background necessary to read and understand the discussed topics is introduced.

The purpose of Chapter 3, which is based on [13], is to answer the following question: does the square of a code "typically" fill the whole space? We give a positive answer, for codes of dimension k and length roughly $k^2/2$ or smaller. Moreover, the convergence speed is exponential if the difference $k(k+1)/2 - n$ is at least linear in k. The proof uses random coding and combinatorial arguments, together with algebraic tools involving the precise computation of the number of quadratic forms of a given rank, and the number of their zeros. As a consequence of this work, it is impossible to rely on random codes in situations where properties of the code square are required, as it will be the full space, hence trivial, with high probability. This impacts for instance secret sharing: it is known [20] that linear, non-multiplicative secret sharing schemes with optimal privacy and reconstruction parameters can be constructed using

random codes; however, due to the results of Chapter 3, such schemes will most likely not be arithmetic[14].

In Chapter 4, based on [53], we characterize Product-MDS pairs of linear codes, i.e. pairs of codes C, D whose product under coordinatewise multiplication has maximum possible minimum distance as a function of the code length and the dimensions $\dim C, \dim D$. We prove in particular, for $C = D$, that if the square of the code C has minimum distance at least 2, and (C, C) is a Product-MDS pair, then either C is a generalized Reed-Solomon code, or C is a direct sum of self-dual codes. The proof is based on new coding-theory analogues of classical theorems of additive combinatorics, namely Kneser's and Vosper's Theorems. More recently [1], these techniques have been used to prove that, among all t-strongly multiplicative secret sharing schemes on n players, only Shamir's scheme can achieve the optimal $t = (n-1)/3$.

Chapter 5, based on [12] focuses on a foundational question which is novel to the best of our knowledge. Multiplicative linear secret sharing is a fundamental notion in the area of secure multiparty computation and, since recently, in the area of two-party cryptography as well. In a nutshell, this notion guarantees that "the product of two secrets is obtained as a linear function of the vector consisting of the coordinatewise product of two respective share-vectors". Suppose we abandon the linearity condition and instead require that this product is obtained by some, not-necessarily-linear "product reconstruction function". Is the resulting notion equivalent to multiplicative linear secret sharing? We show the (perhaps somewhat counter-intuitive) result that this relaxed notion is strictly more general. Concretely, fix a finite field as the base field over which linear secret sharing is considered. Then we show there exists an (exotic) linear secret sharing scheme with an unbounded number of players n such that it has t-privacy with $t = \Omega(n)$ and such that it does admit a product reconstruction function, yet this function is necessarily nonlinear. In addition, we determine the minimum number of players for which those exotic schemes exist. Our proof is based on combinatorial arguments involving quadratic forms. It extends to similar separation results for important variations, such as strongly multiplicative secret sharing.

The first section of each chapter is an overview of the contents of the chapter itself.

[14] For completeness, we mention that [20] points out that also random self-dual codes yield secret sharing schemes with optimal privacy and reconstruction parameters. In addition, this schemes are trivially multiplicative: as the inner product of any two codewords is zero, any coordinate of the product word can be expressed as a linear function of the others. However, this construction does not support more general notions of secret sharing, such as those that require larger secrets or that can tolerate an adversary who deviates from the protocol.