Designs, Codes and Cryptography (2025) 93:4545–4565
https://doi.org/10.1007/s10623-025-01689-7

Distributed matrix multiplication with straggler tolerance over very small fields

Adrián Fidalgo-Díaz[1] · **Umberto Martínez-Peñas**[1]

Received: 3 December 2024 / Revised: 29 May 2025 / Accepted: 1 July 2025 /
Published online: 18 July 2025

Abstract
The problem of distributed matrix multiplication with straggler tolerance over finite fields is considered, focusing on field sizes for which previous solutions were not applicable (for instance, the field of two elements). We employ Reed-Muller-type codes for explicitly constructing the desired algorithms and study their parameters by translating the problem into a combinatorial problem involving sums of discrete convex sets. We generalize polynomial codes and matdot codes, discussing the impossibility of the latter being applicable for very small field sizes, while providing optimal solutions for some regimes of parameters in both cases.

Keywords Distributed matrix multiplication · Footprint bound · Reed–Muller codes · Hyperbolic codes · Minkowski sum

1 Introduction

1.1 Statement of the problem

Heavy computations require significant time to execute. One option for improving execution times is to develop better computers with more advanced CPUs. Unfortunately, this solution becomes increasingly challenging each year, as we seem to be approaching a fundamental limit in clock speed. Traditionally, the alternative has been to improve not the infrastructure but the algorithms themselves, for instance, by incorporating parallelization. A parallelizable algorithm divides a task into smaller sub-tasks that can be computed simultaneously. When implementing these algorithms, a typical approach is to distribute the smaller tasks across multiple computers (worker nodes) that operate independently. Once these computations are completed, a master node collects the results and reconstructs the original computation.

Communicated by D. Panario.

✉ Adrián Fidalgo-Díaz
adrian.fidalgo22@uva.es

✉ Umberto Martínez-Peñas
umberto.martinez@uva.es

[1] IMUVa-Mathematics Research Institute, University of Valladolid, Paseo Belén, 7, 47011 Valladolid, Spain

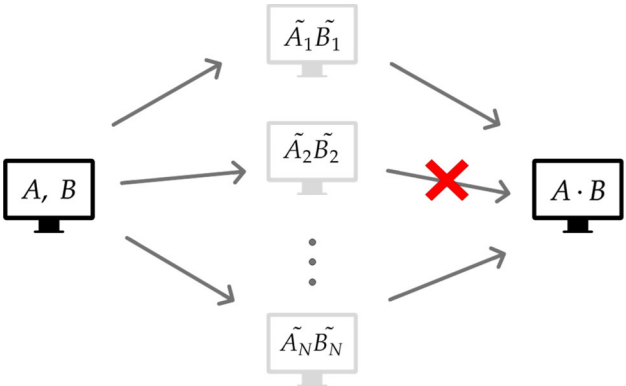

Fig. 1 General scheme of DMM with straggler tolerance, where each $\tilde{A}_i$ and $\tilde{B}_i$ denotes a matrix of lower size than A and B

Roughly speaking, the greater the number of worker nodes, the greater the speedup. Equivalently, having more worker nodes translates into smaller tasks for each one to compute. However, when implementing these algorithms, real-world problems start to arise. If the number of worker nodes is very high, the expected difference between their execution times becomes significant, say because of the network traffic or due to other reasons. This will induce a bottleneck which limits the performance times of the algorithm since the master node must wait for all the worker nodes to complete their tasks in order to obtain the original computation. This effect is often known as the "straggler effect" in the literature. The objetive then is to design algorithms for which the master node can recover the original computation from a subset of worker nodes, considering straggler nodes as non-responsive. Summarizing, we need to recover missing information from the received data, a perfect fit for coding theory methods.

In this manuscript we focus on a computation problem that arises in a large variety of problems: multiplying two matrices. More precisely, we consider the scenario where matrices are defined over a finite field. Among the operations which can be reduced to matrix multiplication over finite fields, we briefly mention attacking code-based cryptography [1–3] (or more generally, application to decoding error-correcting codes [2]) and solving polynomial equations over finite fields via resultants [4, Chapter 3]. The first of these applications is particularly notable when $q = 2$, since McEliece cryptosystem using binary Goppa codes remains one the most promising cryptosystems of this family and, as pointed out in [3], matrix-like operations as Gaussian elimination are the complexity bottleneck of some attacks. For this reason, we believe that, in particular, this kind of distributed algorithms can lead to better attacks in a wide range of cryptosystems taking into account finite fields.

1.2 State of the art and our solution

The most celebrated solution for distributed matrix multiplication with straggler tolerance (DMM, henceforth) was introduced in [5], where the authors proposed encoding both matrices as evaluations of polynomials before multiplying them, or equivalently, encoding them using Reed-Solomon codes. This approach was later generalized in [6], which also employed Reed-Solomon codes but encoded the computations differently. This solution reduces the number of responsive nodes required to recover the original computation, at the cost of higher per-

worker computation and communication costs compared to [5]. So depending on the network specific properties, one of the them will preferred in different scenarios.

Both methods have as downside requiring the number of worker nodes N be smaller than the size of the field over which the matrices are defined. When considering matrices over a finite field of size q, the constraint $N \leq q$ is translated into using at most q worker nodes, which is very restrictive especially for small sizes of q like 2 or 3. Enlarging the field from $\mathbb{F}_q$ to $\mathbb{F}_{q'}$ for some $q' \geq N$ is not a good solution, as this will result in computational overhead. Even worse, if some worker nodes receive a task that can be computed in $\mathbb{F}_q$, they will complete their tasks before those computing in $\mathbb{F}_{q'}$, therefore aggravating the straggler effect. To address this issue, some alternatives deriving from algebraic geometry codes were proposed [7–11], which keep fixed the size of the field while allowing $N \geq q$. For the readers not familiarized with algebraic geometry codes, they work by fixing an algebraic curve over a finite field and evaluating functions on the rational points of the curve in a similar way standard polynomials are evaluated on the elements of the field. For some values of q, remarkably 2 and other non-perfect squares, there exist few options of algebraic curves with many rational points where to evaluate. Consequently, algebraic geometry codes are not an effective solution for these field sizes.

In this work, we propose using a different family of evaluation codes for DMM: codes from multivariate polynomials. More concretely, we propose similar ideas to Reed-Muller codes and hyperbolic codes in order to obtain algorithms for DMM allowing $N \geq q$. Notably, this new approach allows DMM with straggler tolerance over $\mathbb{F}_2$, the field of 2 elements, something that remained impossible with the previous methods of the state of the art.

In a nutshell, the **algorithm** proposed for multiplying two given matrices A and B is:

1. The master node shares with each worker node two small matrices arising as the evaluation of some specific multivariate polynomials which depend on A and B.
2. The worker nodes compute the product of their corresponding matrices (which are smaller than A and B).
3. When enough of the computations finish, the master nodes gathers them and recovers the original matrix multiplication from the smaller products.

2 Preliminaries: the footprint bound

We denote by $\mathbb{F}_q$ the finite field of size q and define the $\mathbb{F}_q$-algebra

$$\mathcal{R} = \frac{\mathbb{F}_q[x_1, x_2, \ldots, x_\ell]}{(x_1^q - x_1, \ldots, x_\ell^q - x_\ell)},$$

where $(\mathcal{A})$ denotes the ideal generated by a set $\mathcal{A}$. Observe that $\mathcal{R}$ is naturally isomorphic to the ring of functions from $\mathbb{F}_q^l$ to $\mathbb{F}_q$ (see Remark 2). Let $\mathcal{V} \subseteq \mathcal{R}$ be an ($\mathbb{F}_q$-linear) vector subspace. For a set $\mathbb{P} \subseteq \mathbb{F}_q^\ell$, consider the evaluation map $ev : \mathcal{V} \to \mathbb{F}_q^{\mathbb{P}}$ which sends $f \in \mathcal{V}$ to $(f(P))_{P \in \mathbb{P}}$, its evaluations in $\mathbb{P}$. Observe that ev is well defined and a linear map.

Notation 1 We write vectors in bold.

For $f \in \mathcal{R} \setminus \{0\}$, we define its restricted **footprint** as $\Delta_{q,\prec}(f) := \{\mathbf{a} \in \mathbb{N}_{<q}^\ell : x^{\mathbf{a}} \notin (\mathrm{LT}(f))\}$, where $\mathbb{N}_{<q} = \{0, 1, \ldots, q-1\}$, $\prec$ is a monomial ordering (defined in $\mathbb{N}^\ell$), $x^{\mathbf{a}} = x_1^{a_1} \cdots x_\ell^{a_\ell}$, for $\mathbf{a} \in \mathbb{N}_{<q}^\ell$, and $\mathrm{LT}(f)$ denotes the leading monomial of f with respect to $\prec$. The following lemma can be found in [12, Sect. IV].

Lemma 1 *([12]) Let $f \in \mathcal{R} \setminus \{0\}$ and $\delta(f) := |\Delta_{q,\prec}(f)|$. If $Z(f)$ denotes the set of zeros of f in $\mathbb{F}_q^l$, then $|Z(f)| \leq \delta(f)$. In particular, if we define $\delta(\mathcal{V}) := \max\{\delta(f) : f \in \mathcal{V} \setminus \{0\}\}$, then $|Z(f)| \leq \delta(\mathcal{V})$, for all $f \in \mathcal{V} \setminus \{0\}$.*

As a consequence, denoting $k := \delta(\mathcal{V})$, if $f \in \mathcal{V}$ and $f(P_i) = 0$ for $k+1$ distinct points $P_1, P_2, \ldots, P_{k+1} \in \mathbb{P}$, then $f = 0$. Equivalently, let $\mathcal{B} := \{f_1, f_2, \ldots, f_\kappa\}$ be a basis of $\mathcal{V}$ and G be the matrix associated to ev in such a basis, that is,

$$G := \begin{pmatrix} f_1(P_1) & f_1(P_2) & \ldots & f_1(P_{k+1}) \\ f_2(P_1) & f_2(P_2) & \ldots & f_2(P_{k+1}) \\ \vdots & \vdots & \ddots & \vdots \\ f_\kappa(P_1) & f_\kappa(P_2) & \ldots & f_\kappa(P_{k+1}) \end{pmatrix}. \tag{1}$$

Then G has a right inverse, i.e., ev is injective.

Remark 1 In general, the bound on the number of common zeros of $\mathcal{V}$ given in Lemma 1 is not an equality. Nevertheless, if $\mathcal{V}$ is generated by a set of monomials closed under divisibility, then the equality holds (see [13, Th. 2.8]). Since this is the case for the majority of vector spaces $\mathcal{V}$ that we consider in this manuscript, the bound of Lemma 1 serves as a good proxy for the number of points needed for interpolation.

3 Multivariate polynomial codes

We present a method for DMM using multivariate polynomials which generalizes polynomial codes [5] and that allows using more than q worker nodes. In fact, the method allows an arbitrary large number of worker nodes for a fixed field size. We give explicit constructions as well as bounds on the recovery threshold (see Sects. 3.2 and 3.3). Let us discuss the framework.

Denote by $\mathbb{F}_q^{r\times s}$ the $\mathbb{F}_q$-vector space of matrices with r rows and s columns with entries in $\mathbb{F}_q$. Let $A \in \mathbb{F}_q^{r\times s}$ and $B \in \mathbb{F}_q^{s\times t}$ be two matrices. In order to multiply them, we split them as block matrices in the following way:

$$\begin{gathered} A := \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_m \end{pmatrix}, \quad B := \begin{pmatrix} B_1 & B_2 & \ldots & B_n \end{pmatrix} \\ \implies AB = \begin{pmatrix} A_1B_1 & A_1B_2 & \ldots & A_1B_n \\ A_2B_1 & A_2B_2 & \ldots & A_2B_n \\ \vdots & \vdots & \ddots & \vdots \\ A_mB_1 & A_mB_2 & \ldots & A_mB_n \end{pmatrix}, \end{gathered} \tag{2}$$

where $A_i \in \mathbb{F}_q^{\frac{r}{m}\times s}$ and $B_i \in \mathbb{F}_q^{s\times\frac{t}{n}}$. We want to find $p_A \in \mathcal{R}^{\frac{r}{m}\times s}$ and $p_B \in \mathcal{R}^{s\times\frac{t}{n}}$ such that

$$p_A := \sum_{i=1}^{m} A_i x^{\mathbf{a}_i}, \quad p_B := \sum_{j=1}^{n} B_j x^{\mathbf{b}_j},$$

and such that, for every $\mathbf{a}, \mathbf{a}' \in D_A := \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ and $\mathbf{b}, \mathbf{b}' \in D_B := \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$,

$$(\mathbf{a}+\mathbf{b})_q = (\mathbf{a}'+\mathbf{b}')_q \implies (\mathbf{a}, \mathbf{b}) = (\mathbf{a}', \mathbf{b}'), \tag{3}$$

where, given $a \in \mathbb{N}_{<2q}$ we define $(a)_q$ as

$$(a)_q := \begin{cases} a & \text{if } a < q, \\ (a \mod q) + 1 & \text{if } q \le a < 2q, \end{cases}$$

and we extend it coordinatewise to $\mathbf{a} \in \mathbb{N}^l$.

Remark 2 Observe that $\mathcal{R}$ is, by definition, the ring of polynomials quotient the following equivalence relation: $f \sim g$ if and only if they define the same function when evaluated in $\mathbb{F}_q^l$. Moreover, the operation $(\cdot)_q$ is the reduction of the exponent of a monomial to its cannonical representation, i.e., $x^{\mathbf{a}}x^{\mathbf{b}} = x^{\mathbf{a}+\mathbf{b}} = x^{(\mathbf{a}+\mathbf{b})_q} \in \mathcal{R}$ for all $\mathbf{a}, \mathbf{b} \in \mathbb{N}_{<q}^{\ell}$.

Next, set $h := p_A p_B \in \mathcal{R}^{\frac{r}{m} \times \frac{t}{n}}$.

Remark 3 By enforcing (3) we ensure that $A_i B_j$ is the coefficient of $x^{\mathbf{a}_i+\mathbf{b}_j}$ in h, so we can retrieve AB from h.

Observe that if $h_{i,j} \in \mathcal{R}$ is the (i, j)th coordinate of h, then $h_{i,j} \in \mathcal{V} := \langle x^{\mathbf{a}+\mathbf{b}} : \mathbf{a} \in D_A, \mathbf{b} \in D_B \rangle_{\mathbb{F}_q}$, where $\langle\cdot\rangle_{\mathbb{F}_q}$ denotes $\mathbb{F}_q$-linear span. So from Sect. 2, we can interpolate h from knowing any $k + 1$ evaluations (the image of h by ev defined with $|\mathbb{P}| = k + 1$), where $k = \delta(\mathcal{V})$, by inverting the matrix G from (1) on the right. This interpolation step has negligible computational complexity compared to that of the matrix multiplication by each worker when the matrix sizes r, s and t are large, as in [7, Sect. 5] (see Appendix A).

This results in the following **algorithm**: the master node shares the evaluations $p_A(P_i) \in \mathbb{F}_q^{\frac{r}{m} \times s}$ and $p_B(P_i) \in \mathbb{F}_q^{s \times \frac{t}{n}}$, with the i-th worker. Then, the workers compute the matrix multiplications $h(P_i) = p_A(P_i)p_B(P_i)$ in parallel. When any $k + 1$ of them end their computations, the master node recovers h by performing componentwise interpolations, and so recovering AB because of Remark 3. In the literature, the number $k + 1$ is called the **recovery threshold**, i.e., the minimum number of responsive nodes needed to recover the product AB.

The remaining part is how to select sets $D_A, D_B \subseteq \mathbb{N}_{<q}^{\ell}$ of sizes $|D_A| = m$ and $|D_B| = n$ satisfying (3) in such way that k is as small as possible. Noticing that $\mathcal{V}$ is generated by monomials, we conclude that

$$k = \delta(\mathcal{V}) = \max\left\{|\Delta(x^{\mathbf{a}})| : x^{\mathbf{a}} \in \mathcal{V}\right\} = q^{\ell} - \min\left\{\prod_{i=1}^{\ell}(q - (a_i)_q) : x^{\mathbf{a}} \in \mathcal{V}\right\}. \quad (4)$$

We summarize the problem of obtaining the coefficients of p_A and p_B satisfying (3) and minimizing (4) in Problem 1.

Notation 2 We define $D_A +_q D_B$ as the Minkowski sum reduced with $(\cdot)_q$, that is

$$D_A +_q D_B := \{(\mathbf{a} + \mathbf{b})_q \in \mathbb{N}_{<q}^{l} : \mathbf{a} \in D_A, \mathbf{b} \in D_B\}.$$

Problem 1 Find $D_A, D_B \subseteq \mathbb{N}_{<q}^{\ell}$ such that

1. $|D_A| = m$ and $|D_B| = n$,
2. $|D_A +_q D_B| = |D_A| \cdot |D_B|$ or, equivalently, satisfying (3), and
3. $\mathrm{FB}(D_A +_q D_B) := \min\{\prod_{i=1}^{\ell}(q - (a_i + b_i)_q) : \mathbf{a} \in D_A, \mathbf{b} \in D_B\}$ is as large as possible.

If D_A and D_B achieve items 1 and 2, we say they are a solution. If item 3 is also achieved, we say they are an optimal solution.

More explicitly, finding a solution for Problem 1 will yield an algorithm for DMM with straggler tolerance for matrices which are subdivided in m and n submatrices as in (2), with recovery threshold $q^l - \mathrm{FB}(D_A +_q D_B) + 1$, as noted in (4). So minimizing the recovery threshold is equivalent to maximizing $\mathrm{FB}(D_A +_q D_B)$. We study such solutions specifying both $\mathrm{FB}(D_A +_q D_B)$ and the recovery threshold for clarity, but the reader should keep in mind that one uniquely determines the other. Even though in Sect. 2 we defined the footprint such that $\delta(f) = q^l - \mathrm{FB}(\mathrm{LT}(f))$, often we will refer to $\mathrm{FB}(D_A + D_B)$ simply as the footprint during the rest of the manuscript.

Remark 4 Notice that the maximum number of worker nodes is q^l, the maximum number of different evaluations we can perform over $\mathbb{F}_q^l$. In the case $\ell = 1$ (polynomials defined over one variable), this algorithm is known as polynomial codes [5] in the literature, and Problem 1 is solved by choosing $D_A := \{0, 1, \ldots, m-1\}$ and $D_B := \{0, m, \ldots, (n-1)m\}$, always under the assumption that $mn < q$. Polynomial codes attain the best information theoretical recovery threshold but the number of worker nodes is limited by the size of the field the matrices are defined over. In particular, small fields such as $\mathbb{F}_2$ or $\mathbb{F}_3$ are not allowed by polynomial codes [5]. Neither constructions arising from algebraic geometry codes [7, 9–11] are applicable for these field sizes because of the lack of algebraic function fields over them.

Remark 5 One of the first things we notice when looking for optimal solutions to Problem 1 is that if D_A (analogously D_B) is not "laying on the axes", then the solution D_A and D_B is not optimal. More formally, if $d_i := \min\{a_i : \mathbf{a} \in D_A\} > 0$, we can consider $D_A' := D_A - (0, \ldots, d_i, \ldots, 0)$ together with D_B, obtaining a strictly better solution in terms of the footprint bound.

3.1 A bound on the recovery threshold

We start by introducing a bound on $\mathrm{FB}(D_A +_q D_B)$ for a solution to Problem 1 and, consequently, a bound on the recovery threshold the algorithm produces. The next definition is [14, Def. 4].

Definition 1 ([14]) Let $q, F, l \in \mathbb{N}$, we define the hyperbolic set $\mathrm{Hyp}_q(F, \ell)$ or simply $\mathrm{Hyp}(F)$ as

$$\mathrm{Hyp}_q(F, \ell) := \left\{ \mathbf{a} \in \mathbb{N}_{<q}^{\ell} : \prod_{i=1}^{l} (q - a_i) \geq F \right\}.$$

The hyperbolic set $\mathrm{Hyp}(F)$ is, by definition, the biggest subset of $\mathbb{N}_{<q}^l$ such that $\mathrm{FB}(\mathrm{Hyp}(F)) \geq F$. Proposition 1 uses this concept to give a bound on $\mathrm{FB}(D_A +_q D_B)$.

Proposition 1 *Consider* $\xi := \max\{F \in \mathbb{N} : |\mathrm{Hyp}(F)| \geq mn\}$. *If* D_A *and* D_B *is a solution, then* $\mathrm{FB}(D_A +_q D_B) \leq \xi$.

Proof Due to the definition of $\mathrm{FB}(D_A +_q D_B)$ we have that $\prod_{i=1}^{\ell}(q - c_i) \geq \mathrm{FB}(D_A +_q D_B)$ for every $\mathbf{c} \in D_A +_q D_B$. Then, $D_A +_q D_B \subseteq \mathrm{Hyp}(\mathrm{FB}(D_A +_q D_B))$. Comparing their sizes we get that $mn \leq |\mathrm{Hyp}(\mathrm{FB}(D_A +_q D_B))|$, and the result follows from the definition of ξ.

Despite the fact that the bound of Proposition 1 is not a closed formula, we will see in Corollary 1 that we can easily compute it. Because of this, in the Appendix we will be able to compare the footprints of our constructions with this bound.

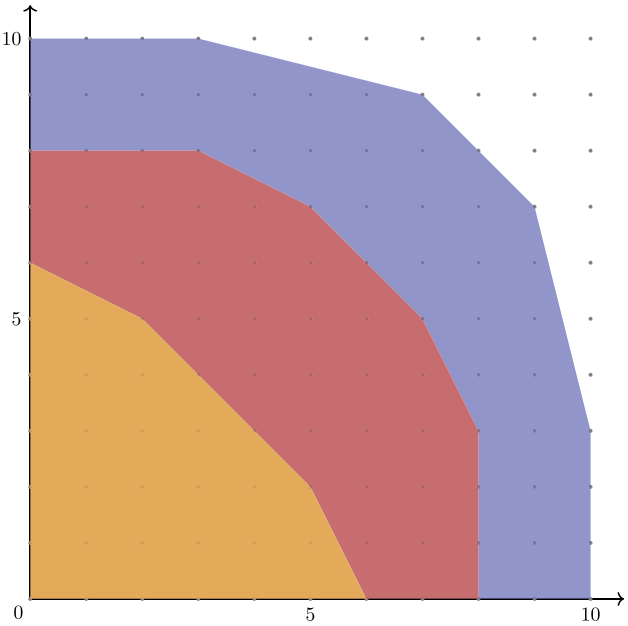

Fig. 2 Hyperbolic sets $\mathrm{Hyp}_{11}(53, 2) \subseteq \mathrm{Hyp}_{11}(24, 2) \subseteq \mathrm{Hyp}_{11}(8, 2)$ represented in yellow, red and blue, respectively

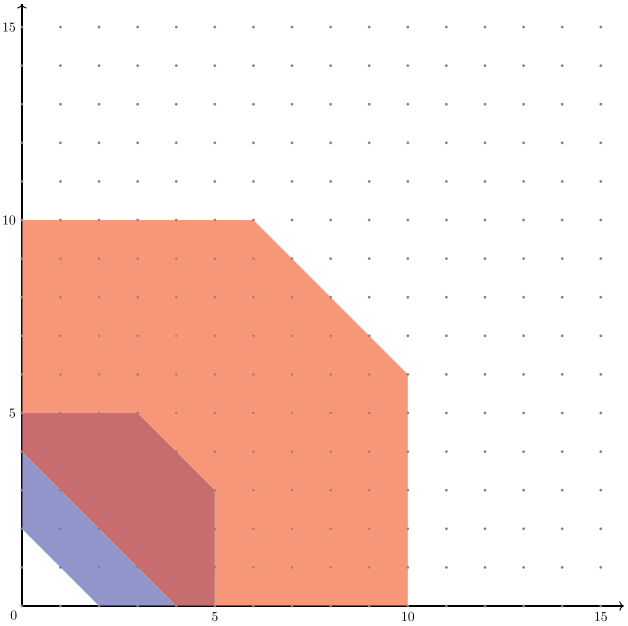

Fig. 3 .

Now we introduce different solutions to Problem 1. We construct the solutions and study the parameters in general, but we group them according to on which regime of the parameters q and l they yield better results. Observe that there is a tradeoff between these two parameters: more variables lead to worse solutions but they are needed when working over fields of small size. This is, either q or l have to be sufficiently large.

3.2 Few variables and moderately small field size

We start studying constructions to solve Problem 1 that behave well for small l and moderate q. As a naive generalization of the chosen degrees in polynomial codes [5], we introduce Construction 1.

Notation 3 Let $\mathbf{s}, \mathbf{k} \in \mathbb{N}^l$, we write $\mathbf{k} * \mathbf{s} := (k_1 s_1, k_2 s_2, \ldots, k_l s_l)$. If $S \subseteq \mathbb{N}^l$, then we write $\mathbf{k} * S := \{\mathbf{k} * \mathbf{s} \in \mathbb{N}^l : \ \mathbf{s} \in S\}$. This operation, sometimes denoted by $\dot{*}$, is often referred to as the Schur product or the star product [15].

Construction 1 (Box) Let $\mathbf{m}, \mathbf{n} \in \mathbb{N}^\ell_{<q}$ such that $m_i n_i \leq q$ for every $i = 1, 2, \ldots, l$. Define

$$D_A := \{\mathbf{a} \in \mathbb{N}^l_{<q} : \ a_i < m_i\},$$
$$D_B := \{\mathbf{m} * \mathbf{k} \in \mathbb{N}^l_{<q} : \ k_i < n_i\}.$$

Because of $m_i n_i \leq q$, we have that $\mathbf{a} + \mathbf{b} \in \mathbb{N}^l_{<q}$ for every $\mathbf{a} \in D_A$ and $\mathbf{b} \in D_B$, resulting in $D_A +_q D_B = D_A + D_B$ being the classical Minkowski sum. By Euclidean division by m_i, we deduce that $|D_A + D_B| = |D_A||D_B|$. More explicitly, if $\mathbf{a}, \mathbf{a}' \in D_A$ and $\mathbf{m} * \mathbf{k}, \mathbf{m} * \mathbf{k}' \in D_B$, then $a_i + m_i k_i = a_i' + m_i k_i'$ implies that $a_i = a_i'$ and $k_i = k_i'$ since $a_i, a_i' < m_i$, by definition. So D_A and D_B are a solution to Problem 1. Moreover, $|D_A| = \prod_{i=1}^\ell m_i$, $|D_B| = \prod_{i=1}^\ell n_i$ and

$$\mathrm{FB}(D_A + D_B) = \prod_{i=1}^{\ell} (q - m_i + 1 - m_i(n_i - 1)) = \prod_{i=1}^{\ell} (q - m_i n_i + 1),$$

so the recovery threshold is $q^l - \prod_{i=1}^\ell (q - m_i n_i + 1) + 1$.

The idea behind Construction 1 is to pick D_A and D_B' as two hypercubes (boxes) and obtain D_B by expanding D_B' with the coordinatewise multiplication (star product), ensuring that $|D_A + D_B| = |D_A||D_B|$. This construction can be generalized selecting D_A and D_B' not necessarily as boxes, and obtaining D_B with the corresponding expansion as Proposition 2 shows.

Proposition 2 *Let $D_A, D_B' \subseteq \mathbb{N}^l_{<q}$ and $m_i := 1 + \max_{\boldsymbol{a}, \boldsymbol{a}' \in D_A} |a_i - a_i'|$. Consider $D_B := \boldsymbol{m} * D_B'$, where $\boldsymbol{m} := (m_1, m_2, \ldots, m_l)$. If $\max_{(\boldsymbol{a}, \boldsymbol{b}) \in D_A \times D_B} (a_i + b_i) < q$ for every $i = 1, 2, \ldots, l$, then $|D_A +_q D_B| = |D_A||D_B|$, thus D_A and D_B are a solution to Problem 1.*

Proof Let $(\mathbf{a}, \mathbf{b}), (\mathbf{a}', \mathbf{b}') \in D_A \times D_B$ such that $\mathbf{a} + \mathbf{b} = \mathbf{a}' + \mathbf{b}'$, then $|a_i - a_i'| = |b_i - b_i'|$ for every $i = 1, 2, \ldots, l$. If $(\mathbf{a}, \mathbf{b}) \neq (\mathbf{a}', \mathbf{b}')$, then $\mathbf{a} \neq \mathbf{a}'$ and $\mathbf{b} \neq \mathbf{b}'$ and

$$|a_i - a_i'| < m_i \leq |b_i - b_i'|,$$

for some i. This contradiction yields $(\mathbf{a}, \mathbf{b}) = (\mathbf{a}', \mathbf{b}')$, so $|D_A +_q D_B| = |D_A||D_B|$.

The approach of Proposition 2 is expanding D_B' by the limits of the smallest box containing D_A. We can obtain a "better" solution by simply selecting D_A as the whole box delimited by

m. In fact, we can improve this construction as Construction 2 shows, choosing D'_B as large as it can be.

Construction 2 (Better box) Let $\mathbf{m} \in \mathbb{N}^l_{<q}$ and $F \in \mathbb{N}$. Consider the sets

$$D_A := \{\mathbf{a} \in \mathbb{N}^l_{<q} : \ a_i < m_i\},$$
$$D_B := \left\{\mathbf{m} * \mathbf{b} \in \mathbb{N}^l_{<q} : \ \prod_{i=1}^{l}(q - m_i + 1 - m_i b_i) \geq F\right\}.$$

Proposition 2 ensures D_A and D_B to be a solution to Problem 1. Moreover, by the definition of D_B, we have $\mathrm{FB}(D_A + D_B) \geq F$, so the recovery threshold is $k + 1 \leq q^l - F + 1$.

Construction 2 is at least as good as (in the majority of cases is strictly better than) Construction 1, since one can always set F in Construction 2 to be $\mathrm{FB}(D_A + D_B)$ with D_A and D_B as in Construction 1, obtaining a better box solution with at least the same footprint and at least the same sizes as sets D_A and D_B. The only non explicit parameter of Construction 2 is the size of D_B. Proposition 3 gives a useful recurrence for computing it.

Definition 2 Let $\mathbf{m} \in \mathbb{N}^l_{<q}$ and $F \in \mathbb{N}$. Define $q_i := q - m_i + 1$ and

$$D_B(F, l) := \left\{(m_1 b_1, m_2 b_2, \ldots, m_l b_l) \in \mathbb{N}^l_{<q} : \ \prod_{i=1}^{l}(q_i - m_i b_i) \geq F\right\}.$$

In particular, $D_B = D_B(F, l)$ in Construction 2.

Proposition 3 *Let* $\boldsymbol{m} \in \mathbb{N}^l_{<q}$ *and* $F \in \mathbb{N}$. *Then,*

$$|D_B(F, l)| = \begin{cases} \max\{0, \lfloor \frac{q_1 - F}{m_1} \rfloor + 1\} & \text{if } l = 1, \\ \sum_{b=0}^{\lfloor \frac{q_l - 1}{m_l} \rfloor} |D_B(\lceil \frac{F}{q_l - m_l b} \rceil, l - 1)| & \text{if } l > 1. \end{cases}$$

Proof If $l = 1$, then

$$\begin{aligned} |D_B(F, 1)| &= |\{m_1 b_1 \in \mathbb{N}_{<q} : \ q_1 - m_1 b_1 \geq F\}| \\ &= \left|\left\{m_1 b_1 \in \mathbb{N}_{<q} : \ \left\lfloor \frac{q_1 - F}{m_1} \right\rfloor \geq b_1\right\}\right| \\ &= \max\left\{0, \left\lfloor \frac{q_1 - F}{m_1} \right\rfloor + 1\right\}. \end{aligned}$$

If $l > 1$, we partition $D_B(F,l)$ and the sum of the sizes gives us the formula:

$$\begin{aligned}
|D_B(F,l)| &= \left| \bigcup_{b=0}^{\lfloor \frac{q_l-1}{m_l} \rfloor} \left\{ (m_1 b_1, \ldots, m_{l-1} b_{l-1}, m_l b) \in \mathbb{N}^l_{<q} \right. \right. \\
&\quad \left. \left. : \prod_{i=1}^{l-1} (q_i - m_i b_i) \geq \left\lceil \frac{F}{q_l - m_l b} \right\rceil \right\} \right| \\
&= \sum_{b=0}^{\lfloor \frac{q_l-1}{m_l} \rfloor} \left| \left\{ (m_1 b_1, \ldots, m_{l-1} b_{l-1}) \in \mathbb{N}^{l-1}_{<q} : \prod_{i=1}^{l-1} (q_i - m_i b_i) \geq \left\lceil \frac{F}{q_l - m_l b} \right\rceil \right\} \right| \\
&= \sum_{b=0}^{\lfloor \frac{q_l-1}{m_l} \rfloor} \left| D_B \left(\left\lceil \frac{F}{q_l - m_l b} \right\rceil, l-1 \right) \right|,
\end{aligned}$$

where the partition is taken by grouping the elements of $D_B(F,l)$ which have the same l-th coordinate.

In Proposition 3, the definition of $D_B(F,l)$ recovers that of $\mathrm{Hyp}_q(F,l)$, in the sense that $D_B(F,l) = \mathrm{Hyp}(F,l)$ by making $\mathbf{m} = (1,1,\ldots,1)$. Moreover, observe that the proof can be adapted to compute the set $D_B(F,l)$ itself, not only its size, by using a backtracking-like algorithm, for example.

As a corollary of Proposition 3 we derive the result from [16] that computes the size of hyperbolic sets. The motivation of this result is to compute the bound on the recovery threshold of Proposition 1. We state it in Corollary 1.

Corollary 1 *([16]) Let $F \geq 1$. Then*

$$|\mathrm{Hyp}_q(F,\ell)| = \begin{cases} \max\{0, q-F+1\} & \text{if } \ell = 1 \\ \sum_{i=1}^{q} |\mathrm{Hyp}_q(\lceil \frac{F}{i} \rceil, \ell-1)| & \text{if } \ell > 1. \end{cases}$$

Proof By setting $i = q - b$ and $\mathbf{m} = (1,1,\ldots,1)$, the results follows as a particular case of Proposition 3.

When restricting to $l = 2$, only two variables, observe that we obtain simpler formulas for computing the size of both the set D_B in Construction 2 and the set $\mathrm{Hyp}_q(F)$.

Corollary 2 *Let $l = 2$ and $\boldsymbol{m} \in \mathbb{N}^2_{<q}$. Then*

$$|D_B(F,2)| = \sum_{b=0}^{\lfloor \frac{q_2-1}{m_2} \rfloor} \max \left\{ 0, \left\lfloor \frac{q_1 - \left\lceil \frac{F}{q_2 - m_2 b} \right\rceil}{m_1} \right\rfloor + 1 \right\}.$$

Regarding hyperbolic sets, for $F \geq 1$ we have

$$|\mathrm{Hyp}_q(F,2)| = \sum_{k=1}^{q} \max \left\{ 0, q - \left\lceil \frac{F}{k} \right\rceil + 1 \right\}.$$

Proof Immediate from $l = 2$ in Proposition 3 and Corollary 1, respectively.

Tables 2 and 3 in the Appendix contain the parameters for several examples of Constructions 1 and 2 when defined over two variables together with the bound given by Proposition 1.

3.3 Many variables and small field size

Now we give a solution to Problem 1 that works well when multiplying matrices over a very small field, that is very small q. We state this solution in Construction 3.

Construction 3 (Separation of variables) Let $\ell = m' + n'$, $0 \leq F_A \leq m'$ and $0 \leq F_B \leq n'$. Define

$$D_A := \{(a_1, a_2, \ldots, a_{m'}, 0, \ldots, 0) \in \mathbb{N}^{\ell}_{<q} : \ \mathbf{a} \in \mathrm{Hyp}_q(F_A, m')\},$$
$$D_B := \{(0, \ldots, 0, b_{m'+1}, b_{m'+2}, \ldots, b_{\ell}) \in \mathbb{N}^{\ell}_{<q} : \ \mathbf{b} \in \mathrm{Hyp}_q(F_B, n')\}.$$

We have that $|D_A + D_B| = |D_A||D_B|$ and, from Definition 1, we obtain the bound

$$\mathrm{FB}(D_A + D_B) = \min\left\{\prod_{i=1}^{m'}(q - a_i) : \ \mathbf{a} \in D_A\right\} \min\left\{\prod_{i=1}^{n'}(q - b_{m'+i}) : \ \mathbf{b} \in D_B\right\} \geq F_A F_B,$$

yielding a recovery threshold of $k+1 \leq q^l - F_A F_B + 1$. Notice that $|D_A| = |\mathrm{Hyp}_q(F_A, m')|$ and $|D_B| = |\mathrm{Hyp}_q(F_B, n')|$. This is the reason for selecting D_A and D_B from hyperbolic sets, because, by Definition 1, this maximizes their sizes when using this method of separation of variables.

Remark 6 Construction 3 may seem naive, but in general it is the best option when considering very small values of q, like 2, 3 or 5. For these q, Construction 2 (and also Construction 1) has very restricted parameters. For example, for $q = 2$, the box $\mathbb{N}^{l}_{<q}$ has only two elements in each coordinate, so in Construction 2, $\mathbf{m}$ is a vector consisting of 0s and 1s. If $m_i = 0$ for some i, then D_A is empty. So $\mathbf{m}$ has to be $(1, 1, \ldots, 1)$, but then D_A consists of only one element, $(0, 0, \ldots, 0)$. We conclude that, for $q = 2$, we are restricted in Construction 2 to $m = 1$, that is, performing DMM only by splitting matrix B and not A, in other words, "introducing redundancy only in one of the matrices". This strategy of encoding only one of the operands was one of the first ideas of the coded computation schemes [17], and presents the disadvantage of performing bad in terms of the communication cost. Observe as well that for $q = 2$ and $l = 2$, we have $|\mathbb{N}^2_{<2}| = 4$ and so $mn \leq 4$, making evident the necessity of using a large number of variables when the field size is small.

Similar to Construction 2, in Construction 3 the sizes of D_A and D_B are not explicit. Fortunately, this is not a problem when giving examples since we can efficiently compute them by using the recurrence of Corollary 1.

To conclude this section, we focus on the case of matrices defined over $\mathbb{F}_2$. In this extremal scenario, Constructions 1 and 2 are not applicable by Remark 6. In general for $q = 2, mn \leq 2^l$, or equivalently $l \geq \log_2(mn)$, is required. As far as the authors know, there are currently no general methods for performing DMM with straggler tolerance over $\mathbb{F}_2$, apart from the one presented in this subsection.

Proposition 4 *Let $F, l \in \mathbb{N}$. Then*

$$|\mathrm{Hyp}_2(F, l)| = \sum_{i=0}^{l - \lceil \log_2 F \rceil} \binom{l}{i}.$$

Proof Let $\mathbf{a} \in \mathbb{N}^l_{<q}$, we denote by $\mathrm{wt}(\mathbf{a})$ the Hamming weight of $\mathbf{a}$, i.e., the number of nonzero coordinates of $\mathbf{a}$. When $q = 2$, we observe that given $\mathbf{a} \in \mathbb{N}^l_{<2} = \{0, 1\}^l$, we have that $\prod_{i=1}^{l}(2 - a_i) = 2^{l-\mathrm{wt}(\mathbf{a})}$. Hence

$$\begin{aligned} |\operatorname{Hyp}_2(F, l)| &= |\{\mathbf{a} \in \{0, 1\}^l : \ 2^{l-\mathrm{wt}(\mathbf{a})} \geq F\}| \\ &= |\{\mathbf{a} \in \{0, 1\}^l : \ \mathrm{wt}(\mathbf{a}) \leq l - \lceil \log_2 F \rceil\}| = \sum_{i=0}^{l-\lceil \log_2 F \rceil} \binom{l}{i}. \end{aligned}$$

Remark 7 Observe that, when $q = 2$, the evaluation code defined by a hyperbolic set is a Reed-Muller code. So separation of variables (Construction 3) in $\mathbb{F}_2$ proposes encoding the matrices A and B using two Reed-Muller codes.

Proposition 4 gives us a way for fast computing the sizes of hyperbolic sets for $q = 2$, and so the sizes of D_A and D_B in Construction 3. In addition, it shows us which are the greatest designed footprints such that $|D_A|$ and $|D_B|$ are maximized: we have to pick $F_1 = 2^\alpha$ and $F_2 = 2^\beta$ for some $\alpha, \beta \in \mathbb{N}$. This motivates the following corollary.

Corollary 3 *Let D_A and D_B as in Construction 3 for $q = 2$. If $F_A = 2^\alpha$ and $F_B = 2^\beta$, then*

$$|D_A| = \sum_{i=0}^{m'-\alpha} \binom{m'}{i} \quad \text{and} \quad |D_B| = \sum_{i=0}^{n'-\beta} \binom{n'}{i}.$$

Proof Straightforward from Proposition 4.

Tables 4 and 5 in the Appendix contain the parameters for several examples of Construction 3 when $q = 2$ together with the bound given by Proposition 1. Tables 6 and 7 contain the parameters for $q > 2$. Observe that, when $q = 2$, Construction 3 is optimal for a large number of non-trivial values of m.

4 Multivariate matdot codes

Now we follow the trail of matdot codes [6], a different method for performing DMM. Matdot codes achieve lower recovery thresholds than polynomial codes [5] at the expense of having a higher communication and computational cost. With the same ideas, in this section we propose polynomials in several variables for DMM as the algorithm presented in Sect. 3 does, but splitting the matrices in a different way. This yields a different algorithm with lower recovery threshold but higher per-woker computation cost (see Appendix A).

As a warmup, consider the following toy example in one variable. Let

$$A = \begin{pmatrix} A_1 & A_2 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix} \tag{5}$$

be two matrices over $\mathbb{F}_q$. Similarly to Sect. 3, A and B are divided in blocks A_1, A_2 and B_1, B_2. respectively. The reader should take into account that this division is not the same as in Sect. 3. Then, consider the polynomials $p_A := A_1 + A_2 x$ and $p_B := B_2 + B_1 x$ and their product, $h := p_A p_B = A_1 B_2 + (A_1 B_1 + A_2 B_2)x + A_2 B_1 x^2$. The key point is that h has AB as the coefficient of the term of degree x, so a similar algorithm can be designed as in Sect. 3 to recover AB from the evaluations of h. This idea of choosing the coefficients of p_A and p_B to "collide in x" is the basis of matdot codes as defined in [6] for general matrix sizes

and one variable. Let us generalize this construction to more variables, which will allow us to use more worker nodes.

Let $A \in \mathbb{F}_q^{r\times s}$ and $B \in \mathbb{F}_q^{s\times t}$, consider the subdivision in blocks of matrices given by

$$A = \begin{pmatrix} A_1 & A_2 & \dots & A_m \end{pmatrix}, \quad B = \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_m \end{pmatrix} \tag{6}$$
$$\implies AB = A_1B_1 + A_2B_2 + \cdots + A_mB_m,$$

where $A_i \in \mathbb{F}_q^{r\times \frac{s}{m}}$ and $B_i \in \mathbb{F}_q^{\frac{s}{m}\times t}$. As in Sect. 3, we define $p_A \in \mathcal{R}^{r\times \frac{s}{m}}$ and $p_B \in \mathcal{R}^{\frac{s}{m}\times t}$ such that

$$p_A := \sum_{i=1}^{m} A_i x^{\mathbf{a}_i}, \quad p_B := \sum_{i=1}^{m} B_i x^{\mathbf{b}_i},$$

but, this time satisfying that, there exist exactly m distinct pairs $(\mathbf{a}, \mathbf{b}) \in D_A \times D_B$, where $D_A := \{\mathbf{a}_1, \dots, \mathbf{a}_m\}$ and $D_B := \{\mathbf{b}_1, \dots, \mathbf{b}_m\}$, such that $\mathbf{d} = (\mathbf{a} + \mathbf{b})_q$ for a fixed $\mathbf{d} \in \mathbb{N}^l_{<q}$. Here, $(\cdot)_q$ denotes the natural sum of exponents of monomials in $\mathcal{R}$, as in Sect. 3. The property is not arbitrary, since it implies AB to be the coefficient of the monomial $x^{\mathbf{d}}$ in $h := p_A p_B$. Using this fact we can design the following **algorithm**.

First, the master node shares $p_A(P_i) \in \mathbb{F}_q^{r\times \frac{s}{m}}$ and $p_B(P_i) \in \mathbb{F}_q^{\frac{s}{m}\times t}$ with the i-th worker node. Then, the worker nodes compute the matrix multiplications $h(P_i) = p_A(P_i)p_B(P_i)$ and give back the result. When enough worker nodes have responded, the master node uses the evaluations $h(P_i)$ to recover h and, consequently, to obtain AB.

The amount of responsive worker nodes necessary to recover h depends on the choice of the sets D_A and D_B, as (4) in Sect. 3 summarizes. The objective then is to minimize $k + 1 := q^l - \min_{(\mathbf{a},\mathbf{b})\in D_A\times D_B}\{\prod_{i=1}^{\ell}(q - (a_i + b_i)_q)\} + 1$, the number of evaluations that ensures we can interpolate h. We express this in Problem 2 in terms of the sum defined in Notation 2.

Problem 2 Find $D_A, D_B \subseteq \mathbb{N}^{\ell}_{<q}$ such that

1. $|D_A| = |D_B| = m$.
2. There exists $\mathbf{d} \in \mathbb{N}^l_{<q}$, such that there are exactly m pairs $(\mathbf{a}_i, \mathbf{b}_i) \in D_A \times D_B$ for $i = 1, 2, \dots, m$ such that $\mathbf{d} = \mathbf{a}_i + \mathbf{b}_i$ and satisfying that $\mathbf{a}_i \neq \mathbf{a}_j$ and $\mathbf{b}_i \neq \mathbf{b}_j$ if $i \neq j$.
3. $\mathrm{FB}(D_A +_q D_B) := \min\{\prod_{i=1}^{\ell}(q - (a_i + b_i)_q) : \ \mathbf{a} \in D_A, \mathbf{b} \in D_B\}$ is as large as possible.

If D_A and D_B achieve items 1 and 2, we say they are a solution. If item 3 is also achieved, we say they are an optimal solution. Sometimes, we will refer to D_A, D_B and $\mathbf{d}$ as the solution itself, even though $\mathbf{d}$ is dependent on the sets D_A and D_B.

Remark 8 In Problem 2, if $\mathbf{d}$ is zero in some coordinate i, that implies that both the elements of D_A and D_B are zero in i. So we can project over $[l] \setminus \{i\}$ and obtain a lower recovery threshold algorithm since it uses fewer variables but maintains the footprint.

We proceed studying different solutions to Problem 2 and their recovery thresholds.

4.1 The box and the case $q = 2$

The simplest construction we can come up with is, as in Sect. 3, choosing a "box" for D_A and D_B. This is Construction 4, which gives a simple way of defining D_A and D_B while generalizing the optimal solution for classical matdot codes.

Construction 4 (Box) Let $\mathbf{m} \in \mathbb{N}^{\ell}_{<q}$ such that $2(m_i - 1) < q$ for every $i = 1, 2, \ldots, l$. Define

$$D_A := D_B = \{\mathbf{a} \in \mathbb{N}^{l}_{<q} : \ a_i < m_i\}.$$

Because of the election of each m_i, we have that $\mathbf{a}+\mathbf{b} \in \mathbb{N}^{l}_{<q}$ for every $\mathbf{a} \in D_A$ and $\mathbf{b} \in D_B$, resulting in $D_A +_q D_B = D_A + D_B$ being the classical Minkowski sum. The rest of the definition ensures D_A and D_B to be a solution to Problem 2, with $\mathbf{d} = (m_1 - 1, m_2 - 1, \ldots, m_l - 1)$. The footprint results to be

$$\mathrm{FB}(D_A + D_B) = \prod_{i=1}^{l}(q - 2m_i + 2).$$

In Construction 4 we have the restriction $2(m_i - 1) < q$ which makes it non applicable for lower values of q. The analog in Sect. 3 was Construction 1, which suffered from the same limitation, in particular for $q = 2$. We were capable of tackling this issue in Sect. 3 by separating variables (see Construction 3), but in the case of multivariate matdot codes, using $q = 2$ is intractable as Proposition 5 shows.

Notation 4 Let $S \subseteq \mathbb{N}^l$, we denote $\mathrm{supp}(S) := \{i \in [l] : \ \exists \mathbf{s} \in S \quad s_i \neq 0\}$.

Proposition 5 *Let D_A, D_B and $\mathbf{d}$ be a solution for Problem 2 with $q = 2$. Then* $\mathrm{FB}(D_A +_q D_B) = 2^{l-|\mathrm{supp}(D_A+_qD_B)|}$. *In particular, if* $\mathrm{supp}(D_A +_q D_B) = [l]$ *(see Remark 8 for its importance), then* $\mathrm{FB}(D_A +_q D_B) = 1$.

Proof Since $d_i = (a_i + b_i)_q$ for some $\mathbf{a} \in D_A$ and $\mathbf{b} \in D_B$, then $d_i = 1$ if $a_i = 1$ or $b_i = 1$, otherwise $d_i = 0$. So $\mathrm{FB}(D_A +_q D_B) = \prod_{i=1}^{l}(2 - d_i) = 2^{l-|\mathrm{supp}(D_A+_qD_B)|}$.

As stated in Proposition 5, there are no methods for DMM with straggler tolerance for matrices over $\mathbb{F}_2$ using multivariate matdot codes (apart from trivial ones). So Construction 3 in Sect. 3.2 still remains the best option for performing DMM over $\mathbb{F}_2$.

Let us study a different way of picking D_A and D_B.

4.2 Optimal solution when $D_A = D_B$

In Construction 4, we picked D_A and D_B as the same set. This is not a requirement to fullfill but both in matdot codes [6] and in AG matdot codes [7], optimality is achieved when doing so. This motivates exploring the idea of restricting to $D_A = D_B$, where the optimality can be satisfactorily studied.

Definition 3 We say $D \subseteq \mathbb{R}^l$ is convex if, for every $\mathbf{a}, \mathbf{a}' \in D$ and $\lambda \in [0, 1] \subseteq \mathbb{R}$, then $\lambda\mathbf{a} + (1 - \lambda)\mathbf{a}' \in D$.

The following result is already known but we do not know any explicit proof in the literature.

Lemma 2 *Consider* $\mathrm{Hyp}_q(F,l)^* := \{\boldsymbol{a} \in \mathbb{R}^l : \prod_{i=1}^{l}(q-a_i) \geq F\}$. *Then,* $\mathrm{Hyp}_q(F,l)^*$ *is convex.*

Proof Let $\mathbf{a}, \mathbf{a}' \in \mathrm{Hyp}_q(F,l)^*$ and $\lambda \in [0,1] \subseteq \mathbb{R}$. Then

$$\prod_{i=1}^{l}(q-\lambda a_i-(1-\lambda)a_i') = \prod_{i=1}^{l}(\lambda(q-a_i)+(1-\lambda)(q-a_i'))$$

Using logarithms, which are known to satisfy that

$$\log(\lambda x+(1-\lambda)y) \geq \lambda\log(x)+(1-\lambda)\log(y),$$

for $x, y > 0$, we obtain:

$$\begin{aligned}
\log\left(\prod_{i=1}^{l}(\lambda(q-a_i)+(1-\lambda)(q-a_i'))\right) &= \sum_{i=1}^{l}\log(\lambda(q-a_i)+(1-\lambda)(q-a_i')) \\
&\geq \sum_{i=1}^{l}(\lambda\log(q-a_i)+(1-\lambda)\log(q-a_i')) \\
&= \lambda\sum_{i=1}^{l}\log(q-a_i)+(1-\lambda)\sum_{i=1}^{l}\log(q-a_i') \\
&= \lambda\log\prod_{i=1}^{l}(q-a_i)+(1-\lambda)\log\prod_{i=1}^{l}(q-a_i') \\
&\geq \lambda\log(F)+(1-\lambda)\log(F) \\
&= \log(F).
\end{aligned}$$

Finally, because the logarithm is an increasing function, we conclude that

$$\prod_{i=1}^{l}(q-\lambda a_i-(1-\lambda)a_i') \geq F.$$

Now, we state Proposition 6 which also can be found in [14, Prop. 4]. Since the proof is short, we give it for completion.

Proposition 6 *[[14, Prop. 4]] Let* $D \subseteq \mathbb{N}^l_{<\frac{q}{2}}$. *Then* $\mathrm{FB}(D+D) \geq F$ *if and only if* $\mathrm{FB}(2\boldsymbol{a}) \geq F$ *for every* $\boldsymbol{a} \in D$.

Proof First, if $\mathrm{FB}(D+D) \geq F$, then $\mathrm{FB}(\mathbf{a}+\mathbf{a}) \geq F$ holds as a particular case. Conversely, suppose that, for every $\mathbf{a} \in D$, $\mathrm{FB}(2\mathbf{a}) \geq F$ holds, which is equivalent to saying that $2\mathbf{a} \in \mathrm{Hyp}_q(F,l)^*$. Let $\mathbf{a}, \mathbf{a}' \in D$ and consider $\mathbf{a}+\mathbf{a}'$. Since $2\mathbf{a}, 2\mathbf{a}' \in \mathrm{Hyp}_q(F,l)^*$, we apply Lemma 2 to conclude that $\mathbf{a}+\mathbf{a}' = \frac{1}{2}(2\mathbf{a})+\frac{1}{2}(2\mathbf{a}') \in \mathrm{Hyp}_q(F,l)^*$, that is $\mathrm{FB}(\mathbf{a}+\mathbf{a}') \geq F$.

Proposition 6 allows us to simplify the computation of $\mathrm{FB}(D_A+D_B)$ when $D_A = D_B$. We exploit this fact and propose Construction 5.

Construction 5 (Half hyperbolic) Let $F \in \mathbb{N}$ and $\mathbf{d} \in \mathbb{N}^l_{<\frac{q}{2}}$. Define

$$D_A := D_B = \{\mathbf{a} \in \mathbb{N}^l_{<\frac{q}{2}} : \mathrm{FB}(2\mathbf{a}) \geq F, \quad \mathrm{FB}(2(\mathbf{d}-\mathbf{a})) \geq F\}.$$

Since $\mathbf{a} \in \mathbb{N}^l_{<\frac{q}{2}}$, $D_A +_q D_B$ is the usual Minkowski sum. We conclude from $\mathbf{a} \in D_A \iff \mathbf{d} - \mathbf{a} \in D_B$ that D_A and D_B are a solution to Problem 2. Moreover, by Proposition 6 we obtain that $\mathrm{FB}(D_A + D_B) \geq F$.

Remark 9 Observe that Construction 5 generalizes Construction 4 by setting $\mathbf{m} = \mathbf{d}$ and $F = 0$.

Proposition 7 *Let $D_A = D_B$ and $\boldsymbol{d}$ be a solution to Problem 2 such that $D_A +_q D_B = D_A + D_B$. Then there exist a solution $D'_A = D'_B$ and $\boldsymbol{d}$ given by Construction 5 such that $|D_A| \leq |D'_A|$ and $\mathrm{FB}(D_A + D_B) \leq \mathrm{FB}(D'_A + D'_B)$. Consequently, if we choose $D \subseteq D'_A$ of size $|D| = |D_A|$, then D and $\boldsymbol{d} - D$ is a solution of size $|D_A|$ with $\mathrm{FB}(D_A + D_B) \leq \mathrm{FB}(D + (\boldsymbol{d} - D))$, thus with smaller or equal recovery threshold.*

Proof Let $F := \mathrm{FB}(D_A + D_B)$. For every $\mathbf{a} \in D_A$ it holds that $\mathrm{FB}(2\mathbf{a}) \geq F$. Consider D'_A and D'_B given by Construction 5 with parameters F and $\mathbf{d}$. Because of Proposition 6, $\mathrm{FB}(D'_A + D'_B) \geq F$ and the statement of the proposition holds.

Proposition 7 shows that Construction 5 yields the best solutions among the ones of the form $D_A = D_B$. So solutions satisfying $D_A = D_B$ can be seen as two subsets of the ones of Construction 5. This easily motivates Construction 5.

Similarly to Proposition 3, the next proposition shows how to compute the size of D_A and D_B recursively in Construction 5. In fact, the same recurrence show us how to compute the sets itselves explicitly. We give only the proof for the sizes for the sake of brevity.

Proposition 8 *Let $F, G \in \mathbb{N}$ and $\boldsymbol{d} \in \mathbb{N}^l_{<q}$. Consider*

$$D(F, G, l) := \left\{ \boldsymbol{a} \in \mathbb{N}^l_{<\frac{q}{2}} : \prod_{i=1}^{l} (q - 2a_i) \geq F, \quad \prod_{i=1}^{l} (q - 2d_i + 2a_i) \geq G \right\}.$$

In particular, $D(F, F, l) = D_A = D_B$ in Construction 5. Then

$$|D(F, G, l)| = \begin{cases} \max\{0, \lfloor \frac{q-F}{2} \rfloor - \lceil \frac{G-q+2d_1}{2} \rceil + 1\} & \text{if } l = 1 \\ \sum_{a=1}^{\lfloor \frac{q-1}{2} \rfloor} \left| D\left(\left\lceil \frac{F}{q-2a} \right\rceil, \left\lceil \frac{G}{q-2d_l+2a} \right\rceil, l-1 \right) \right| & \text{if } l > 1 \end{cases}$$

Proof If $l = 1$, then

$$\begin{aligned} |D(F, G, 1)| &= \left| \left\{ \mathbf{a} \in \mathbb{N}^l_{<\frac{q}{2}} : q - 2a_1 \geq F, \quad q - 2d_1 + 2a_1 \geq G \right\} \right| = \\ &= \left| \left\{ \mathbf{a} \in \mathbb{N}^l_{<\frac{q}{2}} : \left\lfloor \frac{q-F}{2} \right\rfloor \geq a_1 \geq \left\lceil \frac{G-q+2d_1}{2} \right\rceil \right\} \right| = \\ &= \max\left\{ 0, \left\lfloor \frac{q-F}{2} \right\rfloor - \left\lceil \frac{G-q+2d_1}{2} \right\rceil + 1 \right\}. \end{aligned}$$

If $l > 1$, we partition $D(F, G, l)$ depending on the last coordinate of each element and the recursive formula follows easily:

$$\begin{aligned}
|D(F,G,l)| &= \left| \bigcup_{a=0}^{\left\lfloor \frac{q-1}{2} \right\rfloor} \left\{ (a_1, a_2, \ldots, a_{l-1}, a) \in \mathbb{N}^l_{<\frac{q}{2}} : \prod_{i=1}^{l-1} (q - 2a_i) \geq \left\lceil \frac{F}{q-2a} \right\rceil \right. \right. \\
&\quad \left. \left. \prod_{i=1}^{l-1} (q - 2d_i + 2a_i) \geq \left\lceil \frac{G}{q-2d_l+2a} \right\rceil \right\} \right| \\
&= \sum_{a=0}^{\left\lfloor \frac{q-1}{2} \right\rfloor} \left| \left\{ (a_1, a_2, \ldots, a_{l-1}, a) \in \mathbb{N}^l_{<\frac{q}{2}} : \prod_{i=1}^{l-1} (q - 2a_i) \geq \left\lceil \frac{F}{q-2a} \right\rceil \right. \right. \\
&\quad \left. \left. \prod_{i=1}^{l-1} (q - 2d_i + 2a_i) \geq \left\lceil \frac{G}{q-2d_l+2a} \right\rceil \right\} \right| \\
&= \sum_{a=1}^{\left\lfloor \frac{q-1}{2} \right\rfloor} \left| D\left(\left\lceil \frac{F}{q-2a} \right\rceil, \left\lceil \frac{G}{q-2d_l+2a} \right\rceil, l-1 \right) \right|.
\end{aligned}$$

Appendix A Complexity

The complexity analysis is exactly the same as in [7, Sect. 5] where a more detailed treatment of the complexity is given. Alternatively, the reader may check [5] for the original complexity of polynomial codes or [6, Subsect. III B] for that of matdot codes. As in the rest of the manuscript, recall that $k + 1$ denotes the recovery threshold.

In multivariate polynomial codes (respectively, multivariate matdot codes), each worker node has to multiply two matrices of sizes $\frac{r}{m} \times s$ and $s \times \frac{t}{n}$ (resp. $r \times \frac{s}{m}$ and $\frac{s}{m} \times t$). Using the naive matrix multiplication algorithm, this has complexity $\mathcal{O}(\frac{rt}{mn})$ (resp. $\mathcal{O}(\frac{rst}{m})$).

For the decoding process, in multivariate polynomial codes we have to interpolate each entry of $h \in \mathcal{R}^{\frac{r}{m} \times \frac{t}{n}}$. That is, interpolating a polynomial of degree at most k, which has complexity $\mathcal{O}(k^2)$ entrywise. In total, $\mathcal{O}(\frac{rt}{mn}k^2)$. When considering multivariate matdot codes, we only have to interpolate one coordinate of h, having this complexity $\mathcal{O}(rtk)$. For both multivariate polynomial and matdot codes, we have to add the complexity of inverting the matrix which gives the solutions to the linear system of equations for interpolating. This results in complexities $\mathcal{O}(\frac{rt}{mn}k^2 + k^3)$ and $\mathcal{O}(rtk + k^3)$, respectively.

We summarize the complexities in Table 1.

Table 1 Complexity of multivariate polynomial and matdot codes

	Worker computation	Decoding computation
Multivariate polynomial codes	$\mathcal{O}(\frac{rst}{mn})$	$\mathcal{O}(\frac{rt}{mn}k^2 + k^3)$
Multivariate matdot codes	$\mathcal{O}(\frac{rst}{m})$	$\mathcal{O}(rtk + k^3)$

Appendix B Tables

B.1 Multivariate polynomial codes

We present some tables concerning the constructions of Sect. 3. Tables 2 and 3 contain parameters for both Constructions 1 and 2. Tables 4 and 5, for Construction 3.

Given $q = 19, l = 2$, Table 2 shows Construction 1 with $n_i = \lfloor \frac{2q}{3m_i} \rfloor$ and Construction 2 with $|D_B| = \tilde{n}$. The bound of Proposition 1 for each construction are ξ and $\tilde{\xi}$, respectively. The number of worker nodes is $N = 361$ and the recovery threshold, $k + 1$.

Given $q = 25, l = 2$, Table 3 shows Construction 1 with $n_i = \lfloor \frac{2q}{3m_i} \rfloor$ and Construction 2 with $|D_B| = \tilde{n}$. The bound of Proposition 1 for each construction are ξ and $\tilde{\xi}$, respectively. The number of worker nodes is $N = 625$ and the recovery threshold, $k + 1$.

Given $q = 2, l = 10$, Table 4 shows Construction 3 with $m' = 5$, $n' = 5$ and $F_A = F_B = F$. The bound of Proposition 1 is ξ. The number of worker nodes is $N = 1024$ and the recovery threshold, $k + 1$.

Given $q = 2, l = 20$, Table 5 shows Construction 3 with $m' = 10$, $n' = 10$ and $F_A = F_B = F$. The bound of Proposition 1 is ξ. The number of worker nodes is $N = 1048576$ and the recovery threshold, $k + 1$.

Table 2 Example to Constructions 1 and 2

m_i	n_i	m	n	$\tilde{n}$	$\mathrm{FB}(D_A + D_B)$	ξ	$\tilde{\xi}$	$k+1$
1	12	1	144	204	64	102	64	298
2	6	4	36	48	64	102	70	298
3	4	9	16	20	64	102	77	298
4	3	16	9	11	64	102	80	298
5	2	25	4	4	100	143	143	262
6	2	36	4	4	64	102	102	298

Table 3 Example to Constructions 1 and 2

m_i	n_i	m	n	$\tilde{n}$	$\mathrm{FB}(D_A + D_B)$	ξ	$\tilde{\xi}$	$k+1$
1	16	1	256	364	100	168	100	526
2	8	4	64	84	100	168	115	526
3	5	9	25	31	121	192	152	505
4	4	16	16	20	100	168	126	526
5	3	25	9	11	121	192	154	505
6	2	36	4	4	196	270	270	430

Table 4 Example to Construction 3

F	m	$\mathrm{FB}(D_A + D_B)$	ξ	$k+1$
2	31	4	8	1021
4	26	16	16	1009
8	16	64	64	961
16	6	256	256	769

Table 5 Example to Construction 3

F	m	FB$(D_A + D_B)$	ξ	$k+1$
2	1023	4	16	1048573
4	1013	16	64	1048561
8	968	64	128	1048513
16	848	256	512	1048321
32	638	1024	2048	1047553
64	386	4096	4096	1044481
128	176	16384	16384	1032193
256	56	65536	65536	983041
512	11	262144	262144	786433

Table 6 Example to Construction 3

F	m	FB$(D_A + D_B)$	ξ	$k+1$
2	63	4	35	4093
4	61	16	92	4081
8	57	64	236	4033
16	49	256	600	3841
32	33	1024	1540	3073

Table 7 Example to Construction 3

F	m	FB$(D_A + D_B)$	ξ	$k+1$
2	127	4	60	16381
4	125	16	156	16369
8	121	64	396	16321
16	113	256	980	16129
32	97	1024	2440	15361
64	65	4096	6215	12289

Given $q = 64, l = 2$, Table 6 shows Construction 3 with $m' = 1$, $n' = 1$ and $F_A = F_B = F$. The bound of Proposition 1 is ξ. The number of worker nodes is $N = 4096$ and the recovery threshold, $k + 1$.

Given $q = 128, l = 2$, Table 7 shows Construction 3 with $m' = 1$, $n' = 1$ and $F_A = F_B = F$. The bound of Proposition 1 is ξ. The number of worker nodes is $N = 16384$ and the recovery threshold, $k + 1$.

B.2 Multivariate matdot codes

Given $q = 8, l = 3$, Table 8 shows Construction 5 with d computed to maximize m given the footprint F. The number of worker nodes is $N = 512$ and the recovery threshold, $k + 1$.

Given $q = 32, l = 3$, Table 9 shows Construction 5 with d computed to maximize m given the footprint F. The number of worker nodes is $N = 32768$ and the recovery threshold, $k + 1$.

Table 8 Example to Construction 5

F	m	$k+1$
1	64	512
9	62	504
17	56	496
25	50	488
33	38	480
41	38	472
49	26	464
57	26	456

Table 9 Example to Construction 5

F	m	$k+1$
1	4096	32768
513	3044	32256
1025	2106	31744
1537	1440	31232
2049	976	30720
2561	622	30208
3073	374	29696
3585	176	29184

Acknowledgements The authors gratefully acknowledge the support from a María Zambrano contract by the University of Valladolid, Spain (Contract no. E-47-2022-0001486), the support from MCIN/AEI/10.13039/501100011033 and the European Union NextGenerationEU/PRTR (Grant no. TED2021-130358B-I00) and the support from Grant PID2022-138906NB-C21 funded by MICIU/AEI/ 10.13039/501100011033 and by ERDF/EU. Parts of this manuscript have been presented at the Encuentro de Álgebra Computacional y Aplicaciones (EACA), San Lorenzo de El Escorial, Spain, 2024 [18].

Author contributions All authors equally contributed to this manuscript.

Funding Open access funding provided by FEDER European Funds and the Junta de Castilla y León under the Research and Innovation Strategy for Smart Specialization (RIS3) of Castilla y León 2021-2027.

Data availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

1. Aragon, N., Barreto, P., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.-C., Gaborit, P., Ghosh, S., Gueron, S., Güneysu, T., et al.: Bike: bit flipping key encapsulation. HAL hal04278509 (2022)
2. Pellikaan, R., Wu, X.-W., Bulygin, S., Jurrius, R.: Codes, cryptology and curves with computer algebra vol. 1. Cambridge University Press, University Printing house, Cambridge CB2 8BS, United Kingdom (2017)
3. Bernstein, D.J., Lange, T., Peters, C.: Attacking and defending the McEliece cryptosystem. In: Post-Quantum Cryptography: Second International Workshop, PQCrypto 2008 Cincinnati, OH, USA, October 17–19, 2008 Proceedings 2, Springer. pp. 31–46 (2008)
4. Cox D., Little J., O'shea D., Sweedler M.: Ideals, Varieties, and Algorithms, vol. 3, p. 1. Springer, Switzerland (1997).
5. Yu, Q., Maddah-Ali, M., Avestimehr, S.: Polynomial codes: an optimal design for high-dimensional coded matrix multiplication. Adv. Neural Inform. Process. Syst. **30** (2017)
6. Dutta S., Fahim M., Haddadpour F., Jeong H., Cadambe V., Grover P.: On the optimal recovery threshold of coded matrix multiplication. IEEE Trans. Inform. Theory **66**(1), 278–301 (2020).
7. Fidalgo-Díaz, A., Martínez-Peñas, U.: Distributed matrix multiplication with straggler tolerance using algebraic function fields. International Symposium on Information Theory (2024)
8. Machado, R.A., Matthews, G.L., Santos, W.: Hera scheme: secure distributed matrix multiplication via Hermitian codes. In: 2023 IEEE International Symposium on Information Theory (ISIT), IEEE. pp. 1729–1734 (2023)
9. Makkonen, O., Saçıkara, E., Hollanti, C.: Algebraic geometry codes for secure distributed matrix multiplication (2023)
10. Matthews, G.L., Soto, P.: Algebraic geometric rook codes for coded distributed computing. Preprint at arXiv:2405.09746 (2024)
11. Li, J., Li, S., Xing, C.: Algebraic geometry codes for distributed matrix multiplication using local expansions. Preprint at arXiv: 2408.01806 (2024)
12. Geil O., Hoholdt T.: Footprints or generalized Bezout's theorem. IEEE Trans. Inform. Theory **46**(2), 635–641 (2000).
13. Camps, E., López, H., Matthews, G., Sarmiento, E.: Monomial-cartesian codes closed under divisibility. In: Proc. 14th Int. Conf. Finite Fields Appl., pp. 199–208 (2020)
14. García-Marco I., Márquez-Corbella I., Ruano D.: High dimensional affine codes whose square has a designed minimum distance. Des. Codes Cryptogr. **88**(8), 1653–1672 (2020).
15. Randriambololona H.: On products and powers of linear codes under componentwise multiplication. Algorithmic Arith. Geom. Coding Theory **637**, 3–78 (2015).
16. Geil, O., Høholdt, T.: On hyperbolic codes. In: Applied algebra, algebraic algorithms and error-correcting codes: 14th international symposium, AAECC-14 Melbourne, Australia, November 26–30, 2001 Proceedings 14. Springer. pp. 159–171 (2001)
17. Dutta, S., Cadambe, V., Grover, P.: Coded convolution for parallel and distributed computing within a deadline. In: 2017 IEEE International Symposium on Information Theory (ISIT), IEEE. pp. 2403–2407 (2017)
18. Fidalgo-Díaz, A., Martínez-Peñas, U.: Distributed matrix multiplication over small fields with straggler tolerance. In: 2024 Encuentro de Álgebra Computacional Y Aplicaciones (EACA). Interdisciplinary Institute of Mathematics of UCM (2024)