Moulin Coding

A Multilinear-Algebraic Solution for Cloud Storage Services

Hsin-Po WANG
(with Iwan Duursma and Xiao Li)

Department of Mathematics, University of Illinois Urbana-Champaign

2021-8-19 SIAM AG21

Outline

Part I: Motivation from cloud storage services

Part II: Construction of *Moulin Codes*—special case

Part III: Construction of *Moulin Codes*—general case

Outline

Part I: Motivation from cloud storage services

Part II: Construction of *Moulin Codes*—special case

Part III: Construction of *Moulin Codes*—general case

Outline

Part I: Motivation from cloud storage services

Part II: Construction of *Moulin Codes*—special case

Part III: Construction of *Moulin Codes*—general case

Outline

Part I: Motivation from cloud storage services ⇐

Part II: Construction of *Moulin Codes*—special case

Part III: Construction of *Moulin Codes*—general case

Motivation from cloud storage services

hard disk #2 hard disk #4 hard disk #6

hard disk #1 hard disk #3 hard disk #5 hard disk #7

a large file

A cloud storage service is a collection of hard disks that help you store big files.

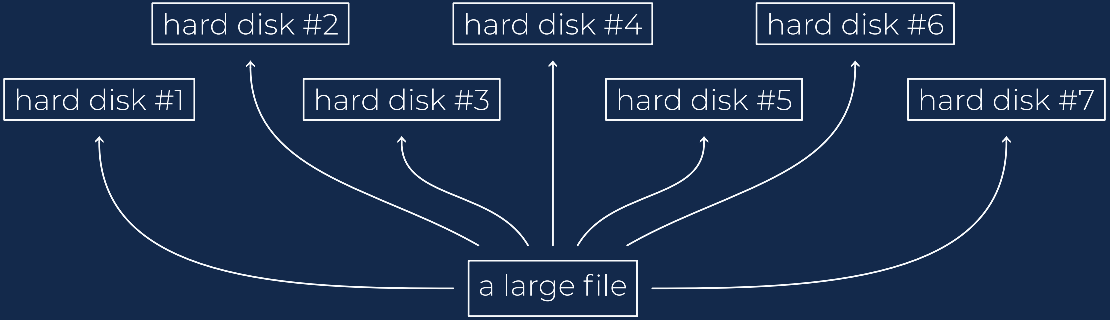
Initially, you upload a big file to the cloud. Each hard disk will store part of the file.

You then delete the local copy to free up some space.

Later, when the file is needed, you download the file from the cloud.
Cloud company guarantees that it is exactly the same file as prevously uploaded.

From your point of view, it just works smoothly. But the cloud sees it differently.

Motivation from cloud storage services



A cloud storage service is a collection of hard disks that help you store big files.

Initially, you upload a big file to the cloud. Each hard disk will store part of the file.

You then delete the local copy to free up some space.

Later, when the file is needed, you download the file from the cloud.
Cloud company guarantees that it is exactly the same file as prevously uploaded.

From your point of view, it just works smoothly. But the cloud sees it differently.

Motivation from cloud storage services

A cloud storage service is a collection of hard disks that help you store big files.

Initially, you upload a big file to the cloud. Each hard disk will store part of the file.

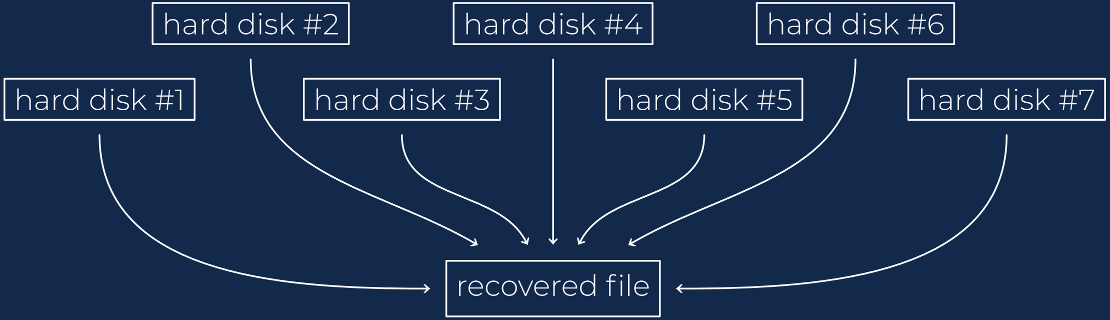
You then delete the local copy to free up some space.

Later, when the file is needed, you download the file from the cloud.

Cloud company guarantees that it is exactly the same file as prevously uploaded.

From your point of view, it just works smoothly. But the cloud sees it differently.

Motivation from cloud storage services



A cloud storage service is a collection of hard disks that help you store big files.

Initially, you upload a big file to the cloud. Each hard disk will store part of the file.

You then delete the local copy to free up some space.

Later, when the file is needed, you download the file from the cloud.
Cloud company guarantees that it is exactly the same file as prevously uploaded.

From your point of view, it just works smoothly. But the cloud sees it differently.

Motivation from cloud storage services

hard disk #2 | hard disk #4 | hard disk #6

hard disk #1 | hard disk #3 | hard disk #5 | hard disk #7

happy customer =)

A cloud storage service is a collection of hard disks that help you store big files.

Initially, you upload a big file to the cloud. Each hard disk will store part of the file.

You then delete the local copy to free up some space.

Later, when the file is needed, you download the file from the cloud.
Cloud company guarantees that it is exactly the same file as prevously uploaded.

From your point of view, it just works smoothly. But the cloud sees it differently.

What could go wrong behind the scene?

hard disk #2 | hard disk #4 | hard disk #6

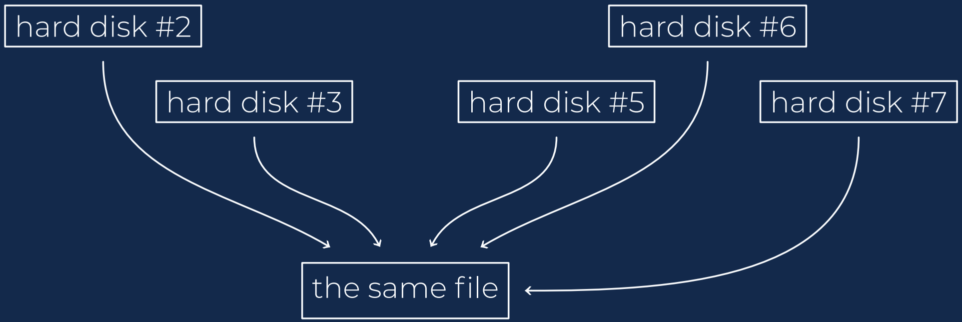
hard disk #1 | hard disk #3 | hard disk #5 | hard disk #7

Difficulty A: Errors (mostly erasures) occur spontaneously. To ensure that errors do not corrupt your file, the cloud is equipped with error-correcting codes.

For instance, we may use a $[7, 5, 3]$-MDS code to protect the file, meaning that every set of 5 disks contains sufficient information to recover the file. Example 2 3

Difficulty B: Fixing errors costs money. Certainly we can recover the file from healthy disks and simulate the uploading phase. Can it be cheaper? Example 2 3

What could go wrong behind the scene?

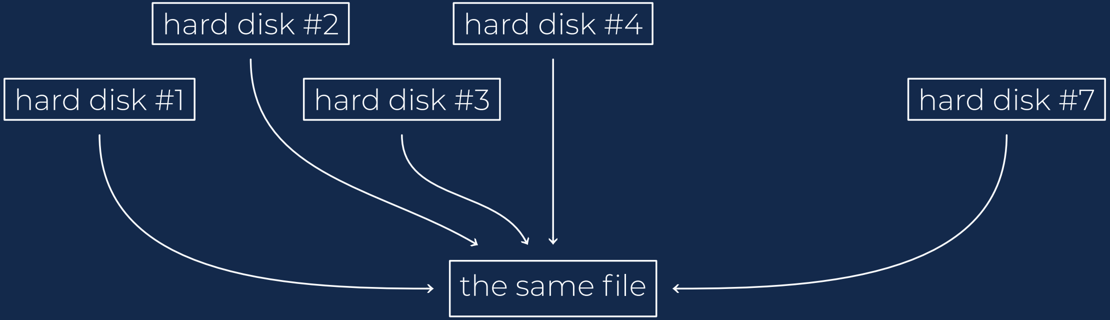


Difficulty A: Errors (mostly erasures) occur spontaneously. To ensure that errors do not corrupt your file, the cloud is equipped with error-correcting codes.

For instance, we may use a $[7,5,3]$-MDS code to protect the file, meaning that every set of 5 disks contains sufficient information to recover the file. Example 2 3

Difficulty B: Fixing errors costs money. Certainly we can recover the file from healthy disks and simulate the uploading phase. Can it be cheaper? Example 2 3

What could go wrong behind the scene?

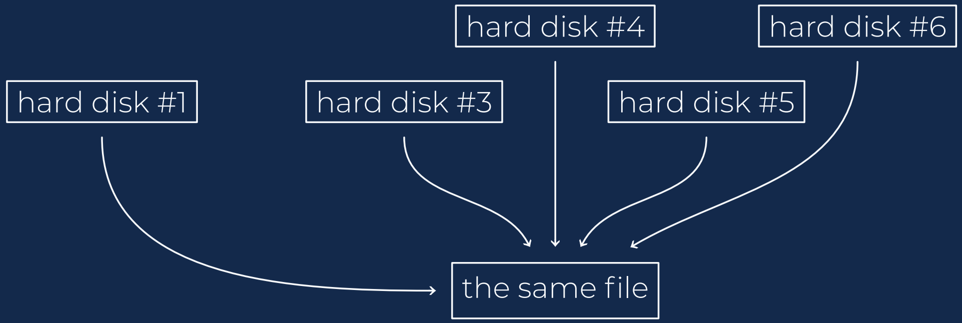


Difficulty A: Errors (mostly erasures) occur spontaneously. To ensure that errors do not corrupt your file, the cloud is equipped with error-correcting codes.

For instance, we may use a $[7, 5, 3]$-MDS code to protect the file, meaning that every set of 5 disks contains sufficient information to recover the file. Example 2 3

Difficulty B: Fixing errors costs money. Certainly we can recover the file from healthy disks and simulate the uploading phase. Can it be cheaper? Example 2 3

What could go wrong behind the scene?

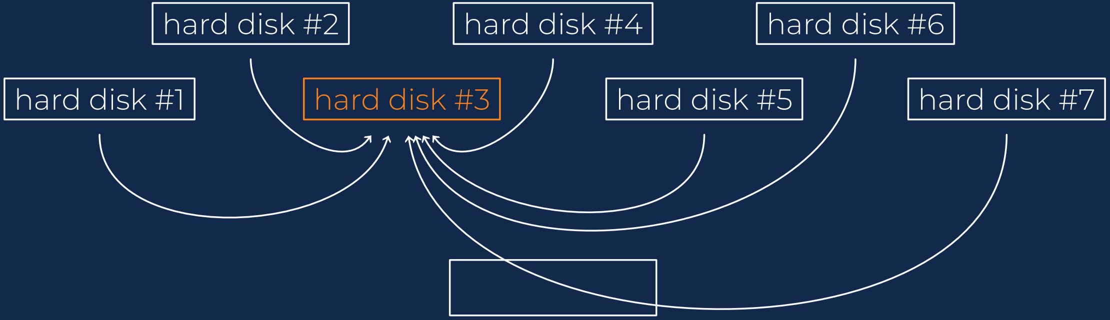


Difficulty A: Errors (mostly erasures) occur spontaneously. To ensure that errors do not corrupt your file, the cloud is equipped with error-correcting codes.

For instance, we may use a $[7, 5, 3]$-MDS code to protect the file, meaning that every set of 5 disks contains sufficient information to recover the file. Example 2 3

Difficulty B: Fixing errors costs money. Certainly we can recover the file from healthy disks and simulate the uploading phase. Can it be cheaper? Example 2 3

What could go wrong behind the scene?

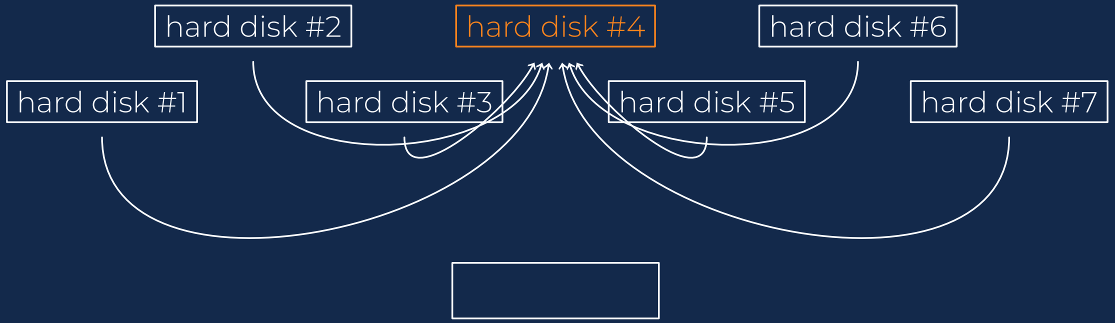


Difficulty A: Errors (mostly erasures) occur spontaneously. To ensure that errors do not corrupt your file, the cloud is equipped with error-correcting codes.

For instance, we may use a $[7, 5, 3]$-MDS code to protect the file, meaning that every set of 5 disks contains sufficient information to recover the file. Example 2 3

Difficulty B: Fixing errors costs money. Certainly we can recover the file from healthy disks and simulate the uploading phase. Can it be cheaper? Example 2 3

What could go wrong behind the scene?

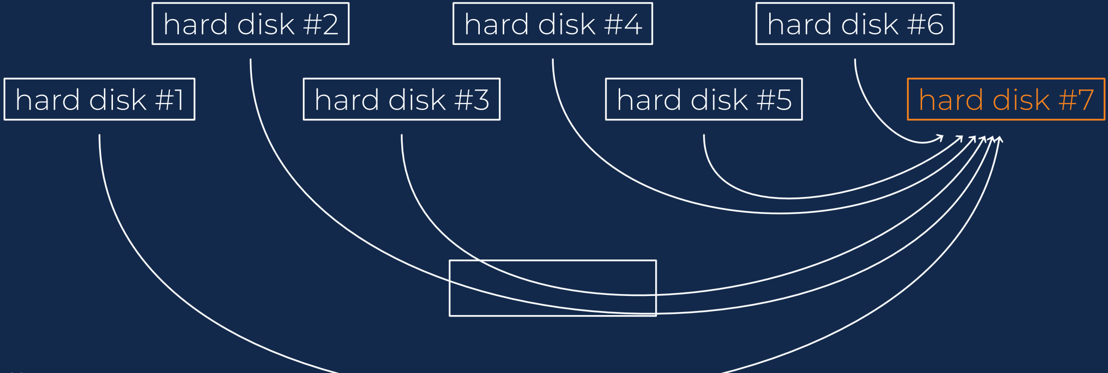


Difficulty A: Errors (mostly erasures) occur spontaneously. To ensure that errors do not corrupt your file, the cloud is equipped with error-correcting codes.

For instance, we may use a $[7,5,3]$-MDS code to protect the file, meaning that every set of 5 disks contains sufficient information to recover the file. Example 2 3

Difficulty B: Fixing errors costs money. Certainly we can recover the file from healthy disks and simulate the uploading phase. Can it be cheaper? Example 2 3

What could go wrong behind the scene?



Difficulty A: Errors (mostly erasures) occur spontaneously. To ensure that errors do not corrupt your file, the cloud is equipped with error-correcting codes.

For instance, we may use a $[7, 5, 3]$-MDS code to protect the file, meaning that every set of 5 disks contains sufficient information to recover the file. Example 2 3

Difficulty B: Fixing errors costs money. Certainly we can recover the file from healthy disks and simulate the uploading phase. Can it be cheaper? Example 2 3

Some notations and requirements of being a good cloud

	size-α disk #2		size-α disk #4		size-α disk #6	
size-α disk #1		size-α disk #3		size-α disk #5		size-α disk #7

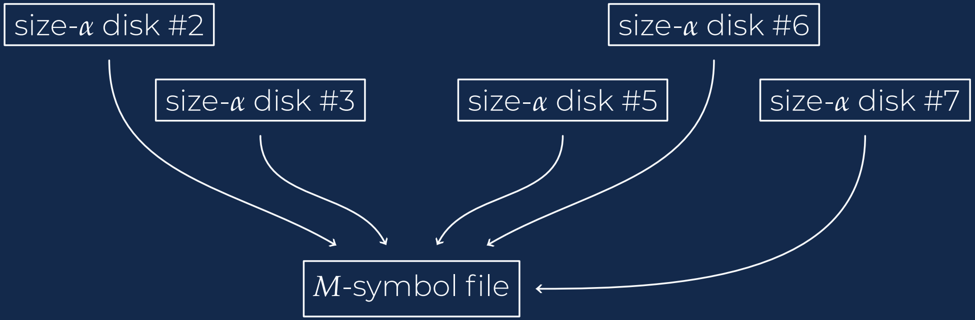
M-symbol file

The following notations are used in literature: The file consists of M symbols. There are n $(= 7)$ hard disks. Every disk stores α symbols. (α called *disk capacity*).

Requirement A: Every set of k $(= 5)$ disks suffices to recover the file.

Requirement B: Every set of d $(= 6)$ disks can reconstruct, from scratch, one other disk by each sending out β symbols of what it has. (β is called *repair bandwidth*.)

Some notations and requirements of being a good cloud

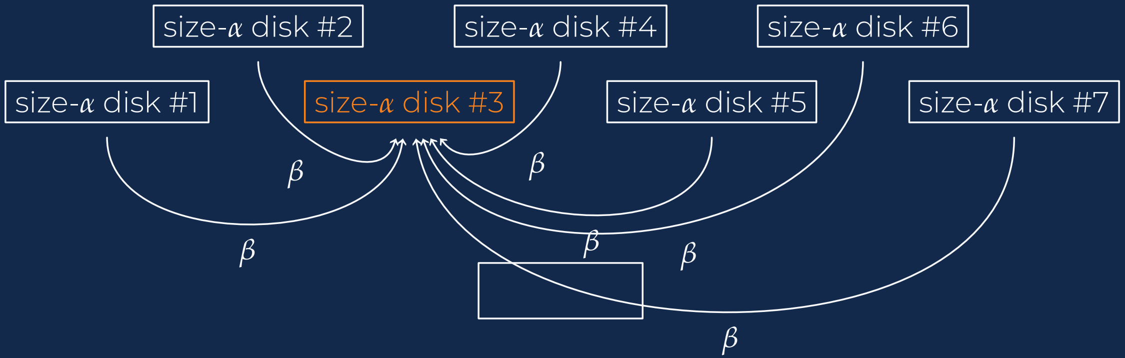


The following notations are used in literature: The file consists of M symbols. There are n $(= 7)$ hard disks. Every disk stores α symbols. (α called *disk capacity*).

Requirement A: Every set of k $(= 5)$ disks suffices to recover the file.

Requirement B: Every set of d $(= 6)$ disks can reconstruct, from scratch, one other disk by each sending out β symbols of what it has. (β is called *repair bandwidth*.)

Some notations and requirements of being a good cloud



The following notations are used in literature: The file consists of M symbols. There are n $(= 7)$ hard disks. Every disk stores α symbols. (α called *disk capacity*).

Requirement A: Every set of k $(= 5)$ disks suffices to recover the file.

Requirement B: Every set of d $(= 6)$ disks can reconstruct, from scratch, one other disk by each sending out β symbols of what it has. (β is called *repair bandwidth*.)

Designing cloud as a linear coding problem

We can now formalize what a cloud should satisfy.

File size M symbols $\longleftrightarrow$ File is a linear map $\varphi\colon \mathbb{F}^M \to \mathbb{F}$, for some finite field $\mathbb{F}$.

n disks, capacity α $\longleftrightarrow$ the hth disk stores $\varphi\restriction_{X_h}$ for some subspace $\mathbb{F}^\alpha \cong X_h \subseteq \mathbb{F}^M$.

Any k disks recover the file $\longleftrightarrow$ Any k subspaces $X_{h_1}, \dots, X_{h_k}$ span φ's domain, $\mathbb{F}^M$.

Any d disks repair, bandwidth β $\longleftrightarrow$ $\exists$ subspaces $\mathbb{F}^\beta \cong Y_h^f \subseteq X_h$ s.t. $Y_{h_1}^f + \cdots + Y_{h_d}^f \supseteq X_f$.

One such tuple $(n, k, d, \alpha, \beta, M; \{X_h\}, \{Y_h^f\})$ is called a *regenerating code*.

Sanity check: $k \le d < n$ (repairing is possible) and $d\beta < M$ (repairing is nontrivial).

Designing cloud as a linear coding problem

We can now formalize what a cloud should satisfy.

File size M symbols $\longleftrightarrow$ File is a linear map $\varphi\colon \mathbb{F}^M \to \mathbb{F}$, for some finite field $\mathbb{F}$.

n disks, capacity $\alpha \longleftrightarrow$ the hth disk stores $\varphi\restriction_{X_h}$ for some subspace $\mathbb{F}^\alpha \cong X_h \subseteq \mathbb{F}^M$.

Any k disks recover the file $\longleftrightarrow$ Any k subspaces $X_{h_1}, \dots, X_{h_k}$ span φ's domain, $\mathbb{F}^M$.

Any d disks repair, bandwidth $\beta \longleftrightarrow \exists$ subspaces $\mathbb{F}^\beta \cong Y_h^f \subseteq X_h$ s.t. $Y_{h_1}^f + \cdots + Y_{h_d}^f \supseteq X_f$.

One such tuple $(n, k, d, \alpha, \beta, M; \{X_h\}, \{Y_h^f\})$ is called a *regenerating code*.

Sanity check: $k \le d < n$ (repairing is possible) and $d\beta < M$ (repairing is nontrivial).

Designing cloud as a linear coding problem

We can now formalize what a cloud should satisfy.

File size M symbols $\longleftrightarrow$ File is a linear map $\varphi\colon \mathbb{F}^M \to \mathbb{F}$, for some finite field $\mathbb{F}$.

n disks, capacity $\alpha \longleftrightarrow$ the hth disk stores $\varphi\restriction_{X_h}$ for some subspace $\mathbb{F}^\alpha \cong X_h \subseteq \mathbb{F}^M$.

Any k disks recover the file $\longleftrightarrow$ Any k subspaces $X_{h_1}, \dots, X_{h_k}$ span φ's domain, $\mathbb{F}^M$.

Any d disks repair, bandwidth $\beta \longleftrightarrow \exists$ subspaces $\mathbb{F}^\beta \cong Y_h^f \subseteq X_h$ s.t. $Y_{h_1}^f + \cdots + Y_{h_d}^f \supseteq X_f$.

One such tuple $(n, k, d, \alpha, \beta, M; \{X_h\}, \{Y_h^f\})$ is called a *regenerating code*.

Sanity check: $k \le d < n$ (repairing is possible) and $d\beta < M$ (repairing is nontrivial).

Designing cloud as a linear coding problem

We can now formalize what a cloud should satisfy.

File size M symbols $\longleftrightarrow$ File is a linear map $\varphi\colon \mathbb{F}^M \to \mathbb{F}$, for some finite field $\mathbb{F}$.

n disks, capacity $\alpha \longleftrightarrow$ the hth disk stores $\varphi\restriction_{X_h}$ for some subspace $\mathbb{F}^\alpha \cong X_h \subseteq \mathbb{F}^M$.

Any k disks recover the file $\longleftrightarrow$ Any k subspaces $X_{h_1}, \dots, X_{h_k}$ span φ's domain, $\mathbb{F}^M$.

Any d disks repair, bandwidth $\beta \longleftrightarrow \exists$ subspaces $\mathbb{F}^\beta \cong Y_h^f \subseteq X_h$ s.t. $Y_{h_1}^f + \cdots + Y_{h_d}^f \supseteq X_f$.

One such tuple $(n, k, d, \alpha, \beta, M; \{X_h\}, \{Y_h^f\})$ is called a *regenerating code*.

Sanity check: $k \le d < n$ (repairing is possible) and $d\beta < M$ (repairing is nontrivial).

Designing cloud as a linear coding problem

We can now formalize what a cloud should satisfy.

File size M symbols $\longleftrightarrow$ File is a linear map $\varphi\colon \mathbb{F}^M \to \mathbb{F}$, for some finite field $\mathbb{F}$.

n disks, capacity α $\longleftrightarrow$ the hth disk stores $\varphi\restriction_{X_h}$ for some subspace $\mathbb{F}^\alpha \cong X_h \subseteq \mathbb{F}^M$.

Any k disks recover the file $\longleftrightarrow$ Any k subspaces $X_{h_1}, \dots, X_{h_k}$ span φ's domain, $\mathbb{F}^M$.

Any d disks repair, bandwidth β $\longleftrightarrow$ $\exists$ subspaces $\mathbb{F}^\beta \cong Y_h^f \subseteq X_h$ s.t. $Y_{h_1}^f + \cdots + Y_{h_d}^f \supseteq X_f$.

One such tuple $(n, k, d, \alpha, \beta, M; \{X_h\}, \{Y_h^f\})$ is called a *regenerating code*.

Sanity check: $k \le d < n$ (repairing is possible) and $d\beta < M$ (repairing is nontrivial).

Designing cloud as a linear coding problem

We can now formalize what a cloud should satisfy.

File size M symbols $\longleftrightarrow$ File is a linear map $\varphi\colon \mathbb{F}^M \to \mathbb{F}$, for some finite field $\mathbb{F}$.

n disks, capacity α $\longleftrightarrow$ the hth disk stores $\varphi\restriction_{X_h}$ for some subspace $\mathbb{F}^\alpha \cong X_h \subseteq \mathbb{F}^M$.

Any k disks recover the file $\longleftrightarrow$ Any k subspaces $X_{h_1}, \dots, X_{h_k}$ span φ's domain, $\mathbb{F}^M$.

Any d disks repair, bandwidth β $\longleftrightarrow$ $\exists$ subspaces $\mathbb{F}^\beta \cong Y_h^f \subseteq X_h$ s.t. $Y_{h_1}^f + \cdots + Y_{h_d}^f \supseteq X_f$.

One such tuple $(n, k, d, \alpha, \beta, M; \{X_h\}, \{Y_h^f\})$ is called a *regenerating code*.

Sanity check: $k \leq d < n$ (repairing is possible) and $d\beta < M$ (repairing is nontrivial).

Designing cloud as a linear coding problem

We can now formalize what a cloud should satisfy.

File size M symbols $\longleftrightarrow$ File is a linear map $\varphi\colon \mathbb{F}^M \to \mathbb{F}$, for some finite field $\mathbb{F}$.

n disks, capacity $\alpha \longleftrightarrow$ the hth disk stores $\varphi\restriction_{X_h}$ for some subspace $\mathbb{F}^\alpha \cong X_h \subseteq \mathbb{F}^M$.

Any k disks recover the file $\longleftrightarrow$ Any k subspaces $X_{h_1}, \dots, X_{h_k}$ span φ's domain, $\mathbb{F}^M$.

Any d disks repair, bandwidth $\beta \longleftrightarrow \exists$ subspaces $\mathbb{F}^\beta \cong Y_h^f \subseteq X_h$ s.t. $Y_{h_1}^f + \cdots + Y_{h_d}^f \supseteq X_f$.

One such tuple $(n, k, d, \alpha, \beta, M; \{X_h\}, \{Y_h^f\})$ is called a *regenerating code*.

Sanity check: $k \leq d < n$ (repairing is possible) and $d\beta < M$ (repairing is nontrivial).

Outline

Part I: Motivation from cloud storage services

Part II: Construction of *Moulin Codes*—special case

Part III: Construction of *Moulin Codes*—general case

Construct Moulin Code for special $k = d$

Recall: n disks; k recover the file; d repair erased disks; $k \le d < n$, otherwise trivial.

Let $W := \mathbb{F}^k$. Consider the wedge-multiplication

$$\begin{aligned} W \otimes W \wedge W &\longrightarrow W \wedge W \wedge W, \\ x \otimes y \wedge z &\longmapsto x \wedge y \wedge z. \end{aligned}$$

It has a natural transpose/dual map, called co-wedge-multiplication

$$\begin{aligned} \nabla \colon W \wedge W \wedge W &\longrightarrow W \otimes W \wedge W, \\ x \wedge y \wedge z &\longmapsto x \otimes y \wedge z - y \otimes x \wedge z + z \otimes x \wedge y. \end{aligned}$$

Let the file be any map $\varphi \colon W \otimes W \wedge W \to \mathbb{F}$ such that $\varphi\restriction_{\operatorname{im}\nabla} = 0$, meaning that it satisfies parity checks $0 = \varphi(\nabla(x \wedge y \wedge z)) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$.

Construct Moulin Code for special $k = d$

Recall: n disks; k recover the file; d repair erased disks; $k \le d < n$, otherwise trivial.

Let $W := \mathbb{F}^k$. Consider the wedge-multiplication

$$\begin{aligned} W \otimes W \wedge W &\longrightarrow W \wedge W \wedge W, \\ x \otimes y \wedge z &\longmapsto x \wedge y \wedge z. \end{aligned}$$

It has a natural transpose/dual map, called co-wedge-multiplication

$$\begin{aligned} \nabla \colon W \wedge W \wedge W &\longrightarrow W \otimes W \wedge W, \\ x \wedge y \wedge z &\longmapsto x \otimes y \wedge z - y \otimes x \wedge z + z \otimes x \wedge y. \end{aligned}$$

Let the file be any map $\varphi \colon W \otimes W \wedge W \to \mathbb{F}$ such that $\varphi\restriction_{\operatorname{im}\nabla} = 0$, meaning that it satisfies parity checks $0 = \varphi(\nabla(x \wedge y \wedge z)) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$.

Construct Moulin Code for special $k = d$

Recall: n disks; k recover the file; d repair erased disks; $k \leq d < n$, otherwise trivial.

Let $W := \mathbb{F}^k$. Consider the wedge-multiplication

$$\begin{aligned} W \otimes W \wedge W &\longrightarrow W \wedge W \wedge W, \\ x \otimes y \wedge z &\longmapsto x \wedge y \wedge z. \end{aligned}$$

It has a natural transpose/dual map, called co-wedge-multiplication

$$\begin{aligned} \nabla \colon W \wedge W \wedge W &\longrightarrow W \otimes W \wedge W, \\ x \wedge y \wedge z &\longmapsto x \otimes y \wedge z - y \otimes x \wedge z + z \otimes x \wedge y. \end{aligned}$$

Let the file be any map $\varphi \colon W \otimes W \wedge W \to \mathbb{F}$ such that $\varphi\restriction_{\operatorname{im} \nabla} = 0$, meaning that it satisfies parity checks $0 = \varphi(\nabla(x \wedge y \wedge z)) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$.

$k = d$ special case, page 2

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$; and $\varphi\restriction_{\operatorname{im}\nabla} = 0$.

Let the n disks choose vectors $c_1, c_2, \ldots, c_n \in W$ that are MDS (i.e., any k span W).
Let the hth disk store the restriction $\varphi\restriction_{c_h \otimes W \wedge W}$

Any k disks recover φ: Let $\mathcal{K}$ be a set of k indices, then by multilinearity & MDSness, $\sum_{h\in\mathcal{K}} c_h \otimes W \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{K}\rangle \otimes W \wedge W = W \otimes W \wedge W =$ the entire domain of φ.

When the fth disk is erased, the hth disk sends it $\varphi\restriction_{c_h \otimes c_f \wedge W}$ to help repair.

Let $\mathcal{D}$ be a set of d indices, $\sum_{h\in\mathcal{D}} c_h \otimes c_f \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{D}\rangle \otimes c_f \wedge W = W \otimes c_f \wedge W$.
We repair $\varphi(c_f \otimes y \wedge z) = \varphi(y \otimes c_f \wedge z) - \varphi(z \otimes c_f \wedge y)$ as RHS is learned from $\varphi\restriction_{W\otimes c_f \wedge W}$

$k = d$ special case, page 2

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$; and $\varphi\restriction_{\operatorname{im}\nabla} = 0$.

Let the n disks choose vectors $c_1, c_2, \dots, c_n \in W$ that are MDS (i.e., any k span W).
Let the hth disk store the restriction $\varphi\restriction_{c_h \otimes W \wedge W}$

Any k disks recover φ: Let $\mathcal{K}$ be a set of k indices, then by multilinearity & MDSness,
$\sum_{h\in\mathcal{K}} c_h \otimes W \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{K}\rangle \otimes W \wedge W = W \otimes W \wedge W =$ the entire domain of φ.

When the fth disk is erased, the hth disk sends it $\varphi\restriction_{c_h \otimes c_f \wedge W}$ to help repair.

Let $\mathcal{D}$ be a set of d indices, $\sum_{h\in\mathcal{D}} c_h \otimes c_f \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{D}\rangle \otimes c_f \wedge W = W \otimes c_f \wedge W$.
We repair $\varphi(c_f \otimes y \wedge z) = \varphi(y \otimes c_f \wedge z) - \varphi(z \otimes c_f \wedge y)$ as RHS is learned from $\varphi\restriction_{W \otimes c_f \wedge W}$

$k = d$ special case, page 2

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$; and $\varphi\restriction_{\operatorname{im}\nabla} = 0$.

Let the n disks choose vectors $c_1, c_2, \ldots, c_n \in W$ that are MDS (i.e., any k span W).
Let the hth disk store the restriction $\varphi\restriction_{c_h \otimes W \wedge W}$

Any k disks recover φ: Let $\mathcal{K}$ be a set of k indices, then by multilinearity & MDSness,
$\sum_{h \in \mathcal{K}} c_h \otimes W \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{K} \rangle \otimes W \wedge W = W \otimes W \wedge W =$ the entire domain of φ.

When the fth disk is erased, the hth disk sends it $\varphi\restriction_{c_h \otimes c_f \wedge W}$ to help repair.

Let $\mathcal{D}$ be a set of d indices, $\sum_{h \in \mathcal{D}} c_h \otimes c_f \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{D} \rangle \otimes c_f \wedge W = W \otimes c_f \wedge W$.
We repair $\varphi(c_f \otimes y \wedge z) = \varphi(y \otimes c_f \wedge z) - \varphi(z \otimes c_f \wedge y)$ as RHS is learned from $\varphi\restriction_{W \otimes c_f \wedge W}$

$k = d$ special case, page 2

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$; and $\varphi\restriction_{\mathrm{im}\,\nabla} = 0$.

Let the n disks choose vectors $c_1, c_2, \dots, c_n \in W$ that are MDS (i.e., any k span W).
Let the hth disk store the restriction $\varphi\restriction_{c_h \otimes W \wedge W}$

Any k disks recover φ: Let $\mathcal{K}$ be a set of k indices, then by multilinearity & MDSness,
$\sum_{h\in\mathcal{K}} c_h \otimes W \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{K}\rangle \otimes W \wedge W = W \otimes W \wedge W =$ the entire domain of φ.

When the fth disk is erased, the hth disk sends it $\varphi\restriction_{c_h \otimes c_f \wedge W}$ to help repair.

Let $\mathcal{D}$ be a set of d indices, $\sum_{h\in\mathcal{D}} c_h \otimes c_f \wedge W = \operatorname{span}\langle c_h : h \in \mathcal{D}\rangle \otimes c_f \wedge W = W \otimes c_f \wedge W$.
We repair $\varphi(c_f \otimes y \wedge z) = \varphi(y \otimes c_f \wedge z) - \varphi(z \otimes c_f \wedge y)$ as RHS is learned from $\varphi\restriction_{W \otimes c_f \wedge W}$

Outline

Part I: Motivation from cloud storage services

Part II: Construction of *Moulin Codes*—special case

Part III: Construction of *Moulin Codes*—general case

Construct Moulin Code for general case $k \leq d$

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$.

Let $V := \mathbb{F}^{d-k}$. Let $U := V \oplus W = \mathbb{F}^d$. Consider the diagram:

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & V \otimes W \wedge W \arrow[d, shift left] \arrow[ddr, "\nabla"] & & V \otimes V \otimes W \arrow[d, shift left] \arrow[ddr, "\nabla"] & & V \otimes V \otimes V \arrow[d, shift left] \\
& & U \otimes W \wedge W \arrow[u, hook, shift left] \arrow[dl, shift left] & & V \otimes U \otimes W \arrow[u, hook, shift left] \arrow[dl, shift left] & & V \otimes V \otimes U \arrow[u, hook, shift left] \arrow[dl, shift left] \\
& W \otimes W \wedge W \arrow[ur, hook, shift left] & & V \otimes W \otimes W \arrow[ur, hook, shift left] & & V \otimes V \otimes W \arrow[ur, hook, shift left] &
\end{tikzcd}
$$

The file is a map φ: direct sum of U-spaces $\to \mathbb{F}$ that vanishes on $\operatorname{im}(\mathrm{id} - \nabla)$.
I.e., $0 =: \varphi(x \wedge y \wedge z) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$ where $x, y, z \in W$
and $\varphi(x \otimes y \wedge z) = \varphi(x \otimes y \otimes z) - \varphi(x \otimes z \otimes y)$ where $x \in V$ and $y, z \in W$
and $\varphi(x \otimes y \otimes z) = \varphi(x \otimes y \otimes z)$ where $x, y \in V$ and $z \in W$ (not tautology but gluing)
and $\varphi(x \otimes y \otimes z) = \varphi(\nabla(x \otimes y \otimes z)) := 0$ where $x, y, z \in V$.

Construct Moulin Code for general case $k \le d$

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$.

Let $V := \mathbb{F}^{d-k}$. Let $U := V \oplus W = \mathbb{F}^d$. Consider the diagram:

$$
\begin{tikzcd}[column sep=tiny]
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, hook] \\
& & U \otimes W \wedge W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes U \otimes W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes V \otimes U \arrow[ur, two heads] \arrow[dl, two heads] & \\
& W \otimes W \wedge W \arrow[ur, hook] & & & V \otimes W \otimes W \arrow[ur, hook] & & & V \otimes V \otimes W \arrow[ur, hook] & &
\end{tikzcd}
$$

The file is a map φ: direct sum of U-spaces $\to \mathbb{F}$ that vanishes on $\operatorname{im}(\mathrm{id} - \nabla)$.
I.e., $0 =: \varphi(x \wedge y \wedge z) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$ where $x, y, z \in W$
and $\varphi(x \otimes y \wedge z) = \varphi(x \otimes y \otimes z) - \varphi(x \otimes z \otimes y)$ where $x \in V$ and $y, z \in W$
and $\varphi(x \otimes y \otimes z) = \varphi(x \otimes y \otimes z)$ where $x, y \in V$ and $z \in W$ (not tautology but gluing)
and $\varphi(x \otimes y \otimes z) = \varphi(\nabla(x \otimes y \otimes z)) := 0$ where $x, y, z \in V$.

Construct Moulin Code for general case $k \le d$

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$.

Let $V := \mathbb{F}^{d-k}$. Let $U := V \oplus W = \mathbb{F}^d$. Consider the diagram:

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, hook, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, hook, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, hook, shift left] \\
& & U \otimes W \wedge W \arrow[ur, two heads, shift left] \arrow[dl, two heads, shift left] & & & V \otimes U \otimes W \arrow[ur, two heads, shift left] \arrow[dl, two heads, shift left] & & & V \otimes V \otimes U \arrow[ur, two heads, shift left] \arrow[dl, two heads, shift left] & \\
& W \otimes W \wedge W \arrow[ur, hook, shift left] & & & V \otimes W \otimes W \arrow[ur, hook, shift left] & & & V \otimes V \otimes W \arrow[ur, hook, shift left] & &
\end{tikzcd}
$$

The file is a map φ: direct sum of U-spaces $\to \mathbb{F}$ that vanishes on $\mathrm{im}(\mathrm{id} - \nabla)$.

I.e., $0 =: \varphi(x \wedge y \wedge z) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$ where $x, y, z \in W$
and $\varphi(x \otimes y \wedge z) = \varphi(x \otimes y \otimes z) - \varphi(x \otimes z \otimes y)$ where $x \in V$ and $y, z \in W$
and $\varphi(x \otimes y \otimes z) = \varphi(x \otimes y \otimes z)$ where $x, y \in V$ and $z \in W$ (not tautology but gluing)
and $\varphi(x \otimes y \otimes z) = \varphi(\nabla(x \otimes y \otimes z)) := 0$ where $x, y, z \in V$.

Construct Moulin Code for general case $k \le d$

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$.

Let $V := \mathbb{F}^{d-k}$. Let $U := V \oplus W = \mathbb{F}^d$. Consider the diagram:

$$
\begin{tikzcd}
W\wedge W\wedge W \arrow[ddr, "\nabla"] & & & V\otimes W\wedge W \arrow[dl, hook, shift left] \arrow[ddr, "\nabla"] & & & V\otimes V\otimes W \arrow[dl, hook, shift left] \arrow[ddr, "\nabla"] & & & V\otimes V\otimes V \arrow[dl, hook, shift left] \\
& & U\otimes W\wedge W \arrow[ur, two heads, shift left] \arrow[dl, two heads, shift left] & & & V\otimes U\otimes W \arrow[ur, two heads, shift left] \arrow[dl, two heads, shift left] & & & V\otimes V\otimes U \arrow[ur, two heads, shift left] \arrow[dl, two heads, shift left] & \\
& W\otimes W\wedge W \arrow[ur, hook, shift left] & & & V\otimes W\otimes W \arrow[ur, hook, shift left] & & & V\otimes V\otimes W \arrow[ur, hook, shift left] & &
\end{tikzcd}
$$

The file is a map φ: direct sum of U-spaces $\to \mathbb{F}$ that vanishes on $\mathrm{im}(\mathrm{id} - \nabla)$.

I.e., $0 =: \varphi(x \wedge y \wedge z) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$ where $x, y, z \in W$

and $\varphi(x \otimes y \wedge z) = \varphi(x \otimes y \otimes z) - \varphi(x \otimes z \otimes y)$ where $x \in V$ and $y, z \in W$

and $\varphi(x \otimes y \otimes z) = \varphi(x \otimes y \otimes z)$ where $x, y \in V$ and $z \in W$ (not tautology but gluing)

and $\varphi(x \otimes y \otimes z) = \varphi(\nabla(x \otimes y \otimes z)) := 0$ where $x, y, z \in V$.

Construct Moulin Code for general case $k \leq d$

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$.

Let $V := \mathbb{F}^{d-k}$. Let $U := V \oplus W = \mathbb{F}^d$. Consider the diagram:

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, two heads, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, two heads, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, two heads, shift left] \\
& & U \otimes W \wedge W \arrow[ur, hook, shift left] \arrow[dl, two heads, shift left] & & & V \otimes U \otimes W \arrow[ur, hook, shift left] \arrow[dl, two heads, shift left] & & & V \otimes V \otimes U \arrow[ur, hook, shift left] \arrow[dl, two heads, shift left] & \\
& W \otimes W \wedge W \arrow[ur, hook, shift left] & & & V \otimes W \otimes W \arrow[ur, hook, shift left] & & & V \otimes V \otimes W \arrow[ur, hook, shift left] & &
\end{tikzcd}
$$

The file is a map φ: direct sum of U-spaces $\to \mathbb{F}$ that vanishes on $\operatorname{im}(\mathrm{id} - \nabla)$.

I.e., $0 =: \varphi(x \wedge y \wedge z) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$ where $x, y, z \in W$

and $\varphi(x \otimes y \wedge z) = \varphi(x \otimes y \otimes z) - \varphi(x \otimes z \otimes y)$ where $x \in V$ and $y, z \in W$

and $\varphi(x \otimes y \otimes z) = \varphi(x \otimes y \otimes z)$ where $x, y \in V$ and $z \in W$ (not tautology but gluing)

and $\varphi(x \otimes y \otimes z) = \varphi(\nabla(x \otimes y \otimes z)) := 0$ where $x, y, z \in V$.

Construct Moulin Code for general case $k \le d$

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$.

Let $V := \mathbb{F}^{d-k}$. Let $U := V \oplus W = \mathbb{F}^d$. Consider the diagram:

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, hook] \\
& & U \otimes W \wedge W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes U \otimes W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes V \otimes U \arrow[ur, two heads] \arrow[dl, two heads] & \\
& W \otimes W \wedge W \arrow[ur, hook] & & & V \otimes W \otimes W \arrow[ur, hook] & & & V \otimes V \otimes W \arrow[ur, hook] & &
\end{tikzcd}
$$

The file is a map φ: direct sum of U-spaces $\to \mathbb{F}$ that vanishes on $\operatorname{im}(\mathrm{id} - \nabla)$.
I.e., $0 =: \varphi(x \wedge y \wedge z) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$ where $x, y, z \in W$
and $\varphi(x \otimes y \wedge z) = \varphi(x \otimes y \otimes z) - \varphi(x \otimes z \otimes y)$ where $x \in V$ and $y, z \in W$
and $\varphi(x \otimes y \otimes z) = \varphi(x \otimes y \otimes z)$ where $x, y \in V$ and $z \in W$ (not tautology but gluing)
and $\varphi(x \otimes y \otimes z) = \varphi(\nabla(x \otimes y \otimes z)) := 0$ where $x, y, z \in V$.

Construct Moulin Code for general case $k \leq d$

Recall: n disks; k recover the file; d repair erased disks. $W := \mathbb{F}^k$.

Let $V := \mathbb{F}^{d-k}$. Let $U := V \oplus W = \mathbb{F}^d$. Consider the diagram:

$$\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, hook] \arrow[ddr, "\nabla"] & \\
& & U \otimes W \wedge W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes U \otimes W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes V \otimes U \arrow[ur, two heads] \arrow[dl, two heads] & & \\
& W \otimes W \wedge W \arrow[ur, hook] & & & V \otimes W \otimes W \arrow[ur, hook] & & & V \otimes V \otimes W \arrow[ur, hook] & & & {}
\end{tikzcd}$$

The file is a map φ: direct sum of U-spaces $\to \mathbb{F}$ that vanishes on $\operatorname{im}(\mathrm{id} - \nabla)$.
I.e., $0 =: \varphi(x \wedge y \wedge z) = \varphi(x \otimes y \wedge z) - \varphi(y \otimes x \wedge z) + \varphi(z \otimes x \wedge y)$ where $x, y, z \in W$
and $\varphi(x \otimes y \wedge z) = \varphi(x \otimes y \otimes z) - \varphi(x \otimes z \otimes y)$ where $x \in V$ and $y, z \in W$
and $\varphi(x \otimes y \otimes z) = \varphi(x \otimes y \otimes z)$ where $x, y \in V$ and $z \in W$ (not tautology but gluing)
and $\varphi(x \otimes y \otimes z) = \varphi(\nabla(x \otimes y \otimes z)) := 0$ where $x, y, z \in V$.

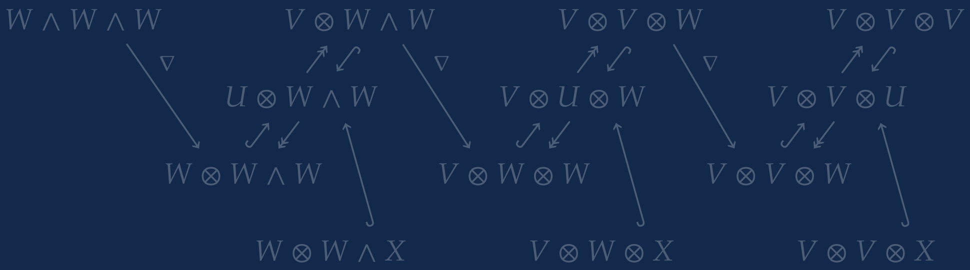
$k \le d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \ldots, a_n \in U$ that are MDS in the sense that (a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X :=$ span$\langle a_h : h \in \mathcal{K} \rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.



$k \le d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that (a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dd, "\nabla"] \arrow[dl, hook] & & V \otimes V \otimes W \arrow[dd, "\nabla"] \arrow[dl, hook] & & V \otimes V \otimes V \arrow[dl, hook] \\
& & U \otimes W \wedge W \arrow[ur, hook] \arrow[dl, hook] & & V \otimes U \otimes W \arrow[ur, hook] \arrow[dl, hook] & & V \otimes V \otimes U \arrow[ur, hook] \arrow[dl, hook] & \\
& W \otimes W \wedge W \arrow[ur, hook] & & V \otimes W \otimes W \arrow[ur, hook] & & V \otimes V \otimes W \arrow[ur, hook] & & \\
& & W \otimes W \wedge X \arrow[uu, hook] & & V \otimes W \otimes X \arrow[uu, hook] & & V \otimes V \otimes X \arrow[uu, hook] &
\end{tikzcd}$$

$k \leq d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \ldots, a_n \in U$ that are MDS in the sense that (a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & & V \otimes W \wedge W \arrow[dll, hook] \arrow[ddr, "\nabla"] & & & & V \otimes V \otimes W \arrow[dll, hook] \arrow[ddr, "\nabla"] & & & & V \otimes V \otimes V \arrow[dll, hook] \\
& & U \otimes W \wedge W \arrow[urr, hook] \arrow[dl, hook] & & & & V \otimes U \otimes W \arrow[urr, hook] \arrow[dl, hook] & & & & V \otimes V \otimes U \arrow[urr, hook] \arrow[dl, hook] & & \\
& W \otimes W \wedge W \arrow[ur, hook] & & & & V \otimes W \otimes W \arrow[ur, hook] & & & & V \otimes V \otimes W \arrow[ur, hook] & & & \\
& & & W \otimes W \wedge X \arrow[uul, hook] & & & & V \otimes W \otimes X \arrow[uul, hook] & & & & V \otimes V \otimes X \arrow[uul, hook] &
\end{tikzcd}
$$

$k \le d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that
(a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X :=$ span$\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & & V \otimes W \wedge W \arrow[dll] \arrow[ddr, "\nabla"] & & & & V \otimes V \otimes W \arrow[dll] \arrow[ddr, "\nabla"] & & & & V \otimes V \otimes V \arrow[dll] \\
& & U \otimes W \wedge W \arrow[urr, hook] \arrow[dl] & & & & V \otimes U \otimes W \arrow[urr, hook] \arrow[dl] & & & & V \otimes V \otimes U \arrow[urr, hook] \arrow[dl] & & \\
& W \otimes W \wedge W \arrow[ur, hook] & & & & V \otimes W \otimes W \arrow[ur, hook] & & & & V \otimes V \otimes W \arrow[ur, hook] & & & \\
& & & W \otimes W \wedge X \arrow[uul, hook] & & & & V \otimes W \otimes X \arrow[uul, hook] & & & & V \otimes V \otimes X \arrow[uul, hook] &
\end{tikzcd}
$$

$k \leq d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File φ: $\bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \ldots, a_n \in U$ that are MDS in the sense that
(a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, shift left] \\
& & U \otimes W \wedge W \arrow[ur, hook, shift left] \arrow[dl, shift left] & & & V \otimes U \otimes W \arrow[ur, hook, shift left] \arrow[dl, shift left] & & & V \otimes V \otimes U \arrow[ur, hook, shift left] \arrow[dl, shift left] & \\
& W \otimes W \wedge W \arrow[ur, hook, shift left] & & & V \otimes W \otimes W \arrow[ur, hook, shift left] & & & V \otimes V \otimes W \arrow[ur, hook, shift left] & & \\
& & & W \otimes W \wedge X \arrow[uul] & & & V \otimes W \otimes X \arrow[uul] & & & V \otimes V \otimes X \arrow[uul]
\end{tikzcd}$$

$k \le d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \ldots, a_n \in U$ that are MDS in the sense that (a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$\begin{tikzcd}
W \wedge W \wedge W \arrow[ddrr, "\nabla"] &&&& V \otimes W \wedge W \arrow[dl, hook, shift left] \arrow[ddrr, "\nabla"] &&&& V \otimes V \otimes W \arrow[dl, hook, shift left] \arrow[ddrr, "\nabla"] &&&& V \otimes V \otimes V \arrow[dl, hook, shift left] \\
&&& U \otimes W \wedge W \arrow[ur, shift left] \arrow[dl, shift left] &&&& V \otimes U \otimes W \arrow[ur, shift left] \arrow[dl, shift left] &&&& V \otimes V \otimes U \arrow[ur, shift left] \arrow[dl, shift left] & \\
&& W \otimes W \wedge W \arrow[ur, hook, shift left] &&&& V \otimes W \otimes W \arrow[ur, hook, shift left] &&&& V \otimes V \otimes W \arrow[ur, hook, shift left] && \\
&&&& W \otimes W \wedge X \arrow[uul, hook] &&&& V \otimes W \otimes X \arrow[uul, hook] &&&& V \otimes V \otimes X \arrow[uul, hook]
\end{tikzcd}$$

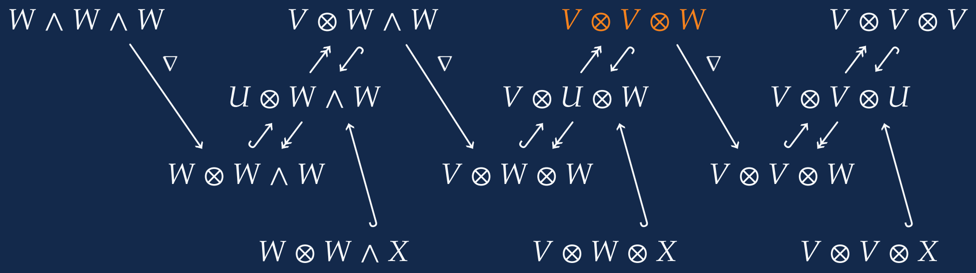
$k \leq d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \ldots, a_n \in U$ that are MDS in the sense that
(a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to
this by parity check, this by downloading, this by MDS, this by projecting.



$k \leq d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that (a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K} \rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & V \otimes W \wedge W \arrow[d, shift left] \arrow[ddr, "\nabla"] & & V \otimes V \otimes W \arrow[d, shift left] \arrow[ddr, "\nabla"] & & V \otimes V \otimes V \arrow[d, shift left] \\
& & U \otimes W \wedge W \arrow[u, hook, shift left] \arrow[dl, shift left] & & V \otimes U \otimes W \arrow[u, hook, shift left] \arrow[dl, shift left] & & V \otimes V \otimes U \arrow[u, hook, shift left] \arrow[dl, shift left] \\
& W \otimes W \wedge W \arrow[ur, hook, shift left] & & V \otimes W \otimes W \arrow[ur, hook, shift left] & & V \otimes V \otimes W \arrow[ur, hook, shift left] & \\
& & W \otimes W \wedge X \arrow[uu, hook] & & {\color{orange} V \otimes W \otimes X} \arrow[uu, hook] & & V \otimes V \otimes X \arrow[uu, hook]
\end{tikzcd}$$

$k \leq d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that
(a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X :=$ span$\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$\begin{tikzcd}
W\wedge W\wedge W \arrow[ddr, "\nabla"] & & & V\otimes W\wedge W \arrow[dl, hook, shift left] \arrow[ddr, "\nabla"] & & & V\otimes V\otimes W \arrow[dl, hook, shift left] \arrow[ddr, "\nabla"] & & & V\otimes V\otimes V \arrow[dl, hook, shift left] \\
& & U\otimes W\wedge W \arrow[ur, shift left] \arrow[dl, shift left] & & & {\color{orange} V\otimes U\otimes W} \arrow[ur, shift left] \arrow[dl, shift left] & & & V\otimes V\otimes U \arrow[ur, shift left] \arrow[dl, shift left] & \\
& W\otimes W\wedge W \arrow[ur, hook, shift left] & & & V\otimes W\otimes W \arrow[ur, hook, shift left] & & & V\otimes V\otimes W \arrow[ur, hook, shift left] & & \\
& & & W\otimes W\wedge X \arrow[uul, hook] & & & V\otimes W\otimes X \arrow[uul, hook] & & & V\otimes V\otimes X \arrow[uul, hook]
\end{tikzcd}$$

$k \leq d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that (a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & V \otimes W \wedge W \arrow[d, shift left] \arrow[ddr, "\nabla"] & & V \otimes V \otimes W \arrow[d, shift left] \arrow[ddr, "\nabla"] & & V \otimes V \otimes V \arrow[d, shift left] \\
& & U \otimes W \wedge W \arrow[u, hook, shift left] \arrow[dl, shift left] & & V \otimes U \otimes W \arrow[u, hook, shift left] \arrow[dl, shift left] & & V \otimes V \otimes U \arrow[u, hook, shift left] \arrow[dl, shift left] \\
& W \otimes W \wedge W \arrow[ur, hook, shift left] & & V \otimes W \otimes W \arrow[ur, hook, shift left] & & V \otimes V \otimes W \arrow[ur, hook, shift left] & \\
& & W \otimes W \wedge X \arrow[uu, hook] & & V \otimes W \otimes X \arrow[uu, hook] & & V \otimes V \otimes X \arrow[uu, hook]
\end{tikzcd}
$$

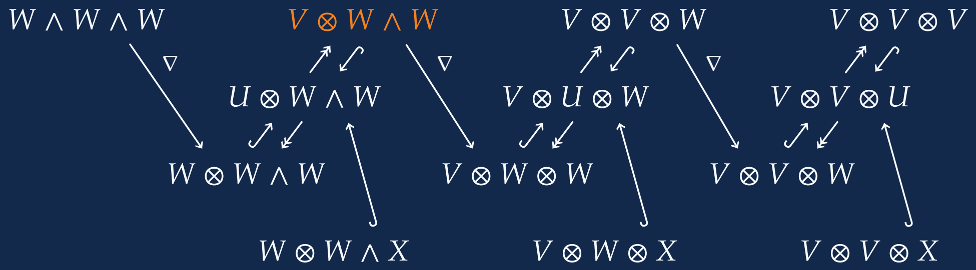
$k \le d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \ldots, a_n \in U$ that are MDS in the sense that
(a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to
this by parity check, this by downloading, this by MDS, this by projecting.



$k \le d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that (a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathscr{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathscr{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$
\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, shift left] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, shift left] \\
& & U \otimes W \wedge W \arrow[ur, hook] \arrow[dl, shift left] & & & V \otimes U \otimes W \arrow[ur, hook] \arrow[dl, shift left] & & & V \otimes V \otimes U \arrow[ur, hook] \arrow[dl, shift left] & \\
& W \otimes W \wedge W \arrow[ur, hook] & & & V \otimes W \otimes W \arrow[ur, hook] & & & V \otimes V \otimes W \arrow[ur, hook] & & \\
& & & W \otimes W \wedge X \arrow[uul, hook] & & & V \otimes W \otimes X \arrow[uul, hook] & & & V \otimes V \otimes X \arrow[uul, hook]
\end{tikzcd}
$$

$k \leq d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that
(a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X :=$ span$\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting.

$$\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & V \otimes W \wedge W \arrow[d, hook] \arrow[ddr, "\nabla"] & & V \otimes V \otimes W \arrow[d, hook] \arrow[ddr, "\nabla"] & & V \otimes V \otimes V \arrow[d, hook] \\
& & U \otimes W \wedge W \arrow[u, shift left] \arrow[dl, shift left] & & V \otimes U \otimes W \arrow[u, shift left] \arrow[dl, shift left] & & V \otimes V \otimes U \arrow[u, shift left] \arrow[dl, shift left] \\
& W \otimes W \wedge W \arrow[ur, hook] & & V \otimes W \otimes W \arrow[ur, hook] & & V \otimes V \otimes W \arrow[ur, hook] & \\
& & W \otimes W \wedge X \arrow[uu, hook] & & V \otimes W \otimes X \arrow[uu, hook] & & V \otimes V \otimes X \arrow[uu, hook]
\end{tikzcd}$$

$k \le d$ general case, page 2

Recall: n disks; k recover the file; d repair erased disks. $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$
File $\varphi\colon \bigoplus$ U-spaces $\to \mathbb{F}$ is such that $\varphi(\text{tensor}) = \varphi(\nabla(\text{tensor}))$.

Let the n disks choose vectors $a_1, a_2, \dots, a_n \in U$ that are MDS in the sense that
(a) every d of them span U; and (b) every k of them span U/W after projection.

Let the hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$ (i.e., replace U by a_h).

Fix k disks $\mathcal{K}$. Let $X := \operatorname{span}\langle a_h : h \in \mathcal{K}\rangle$. Then $V + X = U$. We learn restriction of φ to this by parity check, this by downloading, this by MDS, this by projecting. Done!

$$\begin{tikzcd}
W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \arrow[dl, hook] \\
& & U \otimes W \wedge W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes U \otimes W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes V \otimes U \arrow[ur, two heads] \arrow[dl, two heads] & \\
& W \otimes W \wedge W \arrow[ur, hook] & & & V \otimes W \otimes W \arrow[ur, hook] & & & V \otimes V \otimes W \arrow[ur, hook] & & \\
& & & W \otimes W \wedge X \arrow[uul, hook] & & & V \otimes W \otimes X \arrow[uul, hook] & & & V \otimes V \otimes X \arrow[uul, hook]
\end{tikzcd}$$

$k \le d$ general case, page 3

Recall: $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$. The hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$

When the fth disk is erased, let $a_f =: (b_f, c_f) \in V \oplus W$ and send $\varphi\restriction_{\operatorname{im}\partial}$, where

$$\begin{aligned}\partial\colon V \otimes U &\longrightarrow V \otimes V \otimes U + V \otimes U \otimes W,\\ x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes b_f \otimes y + x \otimes y \otimes c_f,\\ \partial\colon U \otimes W &\longrightarrow V \otimes U \otimes W + U \otimes W \wedge W,\\ x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes c_f \wedge y.\end{aligned}$$

This weird definition is to satisfy the following two properties:

The definition of ∂ extends to tensors of arbitrary length;
it becomes a differential operator (a co-boundary operator) and is linear in a_f.

Let $\nu \in T^pV$ and $\omega \in \Lambda^qW$, then $\varphi(\partial(\nu \otimes \omega)) - \varphi(\partial(\nabla(\nu \otimes \omega))) = (-1)^p \varphi(\nu \otimes a_f \otimes \omega)$.
LHS is learned from the helps from healthy nodes; RHS is the erased content.

$k \leq d$ general case, page 3

Recall: $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$. The hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$

When the fth disk is erased, let $a_f =: (b_f, c_f) \in V \oplus W$ and send $\varphi\restriction_{\operatorname{im} \partial}$, where

$$\begin{aligned}
\partial \colon V \otimes U &\longrightarrow V \otimes V \otimes U + V \otimes U \otimes W, \\
x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes b_f \otimes y + x \otimes y \otimes c_f, \\
\partial \colon U \otimes W &\longrightarrow V \otimes U \otimes W + U \otimes W \wedge W, \\
x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes c_f \wedge y.
\end{aligned}$$

This weird definition is to satisfy the following two properties:

The definition of ∂ extends to tensors of arbitrary length;
it becomes a differential operator (a co-boundary operator) and is linear in a_f.

Let $v \in T^p V$ and $\omega \in \Lambda^q W$, then $\varphi(\partial(v \otimes \omega)) - \varphi(\partial(\nabla(v \otimes \omega))) = (-1)^p \varphi(v \otimes a_f \otimes \omega)$.
LHS is learned from the helps from healthy nodes; RHS is the erased content.

$k \le d$ general case, page 3

Recall: $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$. The hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$

When the fth disk is erased, let $a_f =: (b_f, c_f) \in V \oplus W$ and send $\varphi\restriction_{\operatorname{im} \partial}$, where

$$\begin{aligned} \partial\colon V \otimes U &\longrightarrow V \otimes V \otimes U + V \otimes U \otimes W, \\ x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes b_f \otimes y + x \otimes y \otimes c_f, \\ \partial\colon U \otimes W &\longrightarrow V \otimes U \otimes W + U \otimes W \wedge W, \\ x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes c_f \wedge y. \end{aligned}$$

This weird definition is to satisfy the following two properties:

The definition of ∂ extends to tensors of arbitrary length;
it becomes a differential operator (a co-boundary operator) and is linear in a_f.

Let $v \in T^pV$ and $\omega \in \Lambda^qW$, then $\varphi(\partial(v \otimes \omega)) - \varphi(\partial(\nabla(v \otimes \omega))) = (-1)^p \varphi(v \otimes a_f \otimes \omega)$.
LHS is learned from the helps from healthy nodes; RHS is the erased content.

$k \le d$ general case, page 3

Recall: $(U, V, W) := (\mathbb{F}^d, \mathbb{F}^{d-k}, \mathbb{F}^k)$. The hth disk stores $\varphi\restriction_{a_h \otimes W \wedge W + V \otimes a_h \otimes W + V \otimes V \otimes a_h}$

When the fth disk is erased, let $a_f =: (b_f, c_f) \in V \oplus W$ and send $\varphi\restriction_{\operatorname{im} \partial}$, where

$$\begin{aligned}
\partial \colon V \otimes U &\longrightarrow V \otimes V \otimes U + V \otimes U \otimes W, \\
x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes b_f \otimes y + x \otimes y \otimes c_f, \\
\partial \colon U \otimes W &\longrightarrow V \otimes U \otimes W + U \otimes W \wedge W, \\
x \otimes y &\longmapsto b_f \otimes x \otimes y - x \otimes c_f \wedge y.
\end{aligned}$$

This weird definition is to satisfy the following two properties:

The definition of ∂ extends to tensors of arbitrary length;
it becomes a differential operator (a co-boundary operator) and is linear in a_f.

Let $v \in T^p V$ and $\omega \in \Lambda^q W$, then $\varphi(\partial(v \otimes \omega)) - \varphi(\partial(\nabla(v \otimes \omega))) = (-1)^p \varphi(v \otimes a_f \otimes \omega)$.
LHS is learned from the helps from healthy nodes; RHS is the erased content.

Full generality Moulin Codes with larger diagram

$$
\begin{tikzcd}[column sep=tiny]
W \wedge W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \wedge W \arrow[ddr, "\nabla"] \arrow[dl, hook] & & & V \otimes V \otimes W \wedge W \arrow[ddr, "\nabla"] \arrow[dl, hook] & & & V \otimes V \otimes V \otimes W \arrow[ddr, "\nabla"] \arrow[dl, hook] & & & V \otimes V \otimes V \otimes V \arrow[dl, hook] \\
& & U \otimes W \wedge W \wedge W \arrow[ur, hook] \arrow[dl, hook] & & & V \otimes U \otimes W \wedge W \arrow[ur, hook] \arrow[dl, hook] & & & V \otimes V \otimes U \otimes W \arrow[ur, hook] \arrow[dl, hook] & & & V \otimes V \otimes V \otimes U \arrow[ur, hook] \arrow[dl, hook] & \\
& W \otimes W \wedge W \wedge W \arrow[ur, hook] & & & V \otimes W \otimes W \wedge W \arrow[ur, hook] & & & V \otimes V \otimes W \otimes W \arrow[ur, hook] & & & V \otimes V \otimes V \otimes W \arrow[ur, hook] & &
\end{tikzcd}
$$

$$
\begin{tikzcd}
T^{p-1}V \otimes V \otimes \Lambda^{q+1}W \arrow[ddr, "\nabla"] & & & T^pV \otimes V \otimes \Lambda^q W \arrow[ddr, "\nabla"] \arrow[dl, hook] & \\
& & T^pV \otimes U \otimes \Lambda^q W \arrow[ur, hook] \arrow[dl, hook] & & \\
& T^pV \otimes W \otimes \Lambda^q W \arrow[ur, hook] & & & T^{p+1}V \otimes W \wedge \Lambda^{q-1}W
\end{tikzcd}
$$

Full generality Moulin Codes with larger diagram

$$
\begin{tikzcd}[column sep=tiny]
W \wedge W \wedge W \wedge W \arrow[ddr, "\nabla"] & & & V \otimes W \wedge W \wedge W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes W \wedge W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \otimes W \arrow[dl, hook] \arrow[ddr, "\nabla"] & & & V \otimes V \otimes V \otimes V \arrow[dl, hook] \\
& & U \otimes W \wedge W \wedge W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes U \otimes W \wedge W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes V \otimes U \otimes W \arrow[ur, two heads] \arrow[dl, two heads] & & & V \otimes V \otimes V \otimes U \arrow[ur, two heads] \arrow[dl, two heads] & \\
& W \otimes W \wedge W \wedge W \arrow[ur, hook] & & & V \otimes W \otimes W \wedge W \arrow[ur, hook] & & & V \otimes V \otimes W \otimes W \arrow[ur, hook] & & & V \otimes V \otimes V \otimes W \arrow[ur, hook] & &
\end{tikzcd}
$$

$$
\begin{tikzcd}
T^{p-1}V \otimes V \otimes \Lambda^{q+1}W \arrow[ddr, "\nabla"] & & & T^pV \otimes V \otimes \Lambda^qW \arrow[dl, hook] \arrow[ddr, "\nabla"] & \\
& & T^pV \otimes U \otimes \Lambda^qW \arrow[ur, two heads] \arrow[dl, two heads] & & \\
& T^pV \otimes W \otimes \Lambda^qW \arrow[ur, hook] & & & T^{p+1}V \otimes W \wedge \Lambda^{q-1}W
\end{tikzcd}
$$

Full generality Moulin Codes, page 2

$$\begin{aligned}
\nabla(u \wedge v \wedge w \wedge x \wedge y \wedge z) &= u \otimes v \wedge w \wedge x \wedge y \wedge z \\
&- v \otimes u \wedge w \wedge x \wedge y \wedge z \\
&+ w \otimes u \wedge v \wedge x \wedge y \wedge z \\
&- x \otimes u \wedge v \wedge w \wedge y \wedge z \\
&+ y \otimes u \wedge v \wedge w \wedge x \wedge z \\
&- z \otimes u \wedge v \wedge w \wedge x \wedge y.
\end{aligned}$$

$$\begin{aligned}
\partial(u \otimes v \otimes w \otimes x \otimes y \otimes \zeta) &= b_f \otimes u \otimes v \otimes w \otimes x \otimes y \otimes \zeta \\
&- u \otimes b_f \otimes v \otimes w \otimes x \otimes y \otimes \zeta \\
&+ u \otimes v \otimes b_f \otimes w \otimes x \otimes y \otimes \zeta \\
&- u \otimes v \otimes w \otimes b_f \otimes x \otimes y \otimes \zeta \\
&+ u \otimes v \otimes w \otimes x \otimes b_f \otimes y \otimes \zeta \\
&+ u \otimes v \otimes w \otimes x \otimes y \otimes c_f \wedge \zeta.
\end{aligned}$$

Full generality Moulin Codes, page 2

$$\begin{aligned}\nabla(u \wedge v \wedge w \wedge x \wedge y \wedge z) = u \otimes v \wedge w \wedge x \wedge y \wedge z \\ - v \otimes u \wedge w \wedge x \wedge y \wedge z \\ + w \otimes u \wedge v \wedge x \wedge y \wedge z \\ - x \otimes u \wedge v \wedge w \wedge y \wedge z \\ + y \otimes u \wedge v \wedge w \wedge x \wedge z \\ - z \otimes u \wedge v \wedge w \wedge x \wedge y.\end{aligned}$$

$$\begin{aligned}\partial(u \otimes v \otimes w \otimes x \otimes y \otimes \zeta) = b_f \otimes u \otimes v \otimes w \otimes x \otimes y \otimes \zeta \\ - u \otimes b_f \otimes v \otimes w \otimes x \otimes y \otimes \zeta \\ + u \otimes v \otimes b_f \otimes w \otimes x \otimes y \otimes \zeta \\ - u \otimes v \otimes w \otimes b_f \otimes x \otimes y \otimes \zeta \\ + u \otimes v \otimes w \otimes x \otimes b_f \otimes y \otimes \zeta \\ + u \otimes v \otimes w \otimes x \otimes y \otimes c_f \wedge \zeta.\end{aligned}$$

The performance of regenerating code

Recall: n disks; k recover file; d repair erasure; capacity α; bandwidth β; file size M.

Some reductions for ease of comparison: (n,k,d) depends on actual applications.
So papers on this subject usually fix (n,k,d) and compare different codes' (α,β,M).

Researchers usually start from $n=d+1$ and increase n later,
so it makes sense to ignore n and parametrize the comparison by (k,d).

It is also common to assume that the file is very very large, so
M can be pretty large and we only care about the long-term efficiency $(\alpha/M,\beta/M)$.

In short, it is reasonable to plot all possible $(\alpha/M,\beta/M)$ for each and every (k,d).

The performance of regenerating code

Recall: n disks; k recover file; d repair erasure; capacity α; bandwidth β; file size M.

Some reductions for ease of comparison: (n,k,d) depends on actual applications.
So papers on this subject usually fix (n,k,d) and compare different codes' (α,β,M).

Researchers usually start from $n = d + 1$ and increase n later,
so it makes sense to ignore n and parametrize the comparison by (k,d).

It is also common to assume that the file is very very large, so
M can be pretty large and we only care about the long-term efficiency $(\alpha/M,\beta/M)$.

In short, it is reasonable to plot all possible $(\alpha/M,\beta/M)$ for each and every (k,d).

The performance of regenerating code

Recall: n disks; k recover file; d repair erasure; capacity α; bandwidth β; file size M.

Some reductions for ease of comparison: (n, k, d) depends on actual applications.
So papers on this subject usually fix (n, k, d) and compare different codes' (α, β, M).

Researchers usually start from $n = d + 1$ and increase n later,
so it makes sense to ignore n and parametrize the comparison by (k, d).

It is also common to assume that the file is very very large, so
M can be pretty large and we only care about the long-term efficiency $(\alpha/M, \beta/M)$.

In short, it is reasonable to plot all possible $(\alpha/M, \beta/M)$ for each and every (k, d).

The performance of regenerating code

Recall: n disks; k recover file; d repair erasure; capacity α; bandwidth β; file size M.

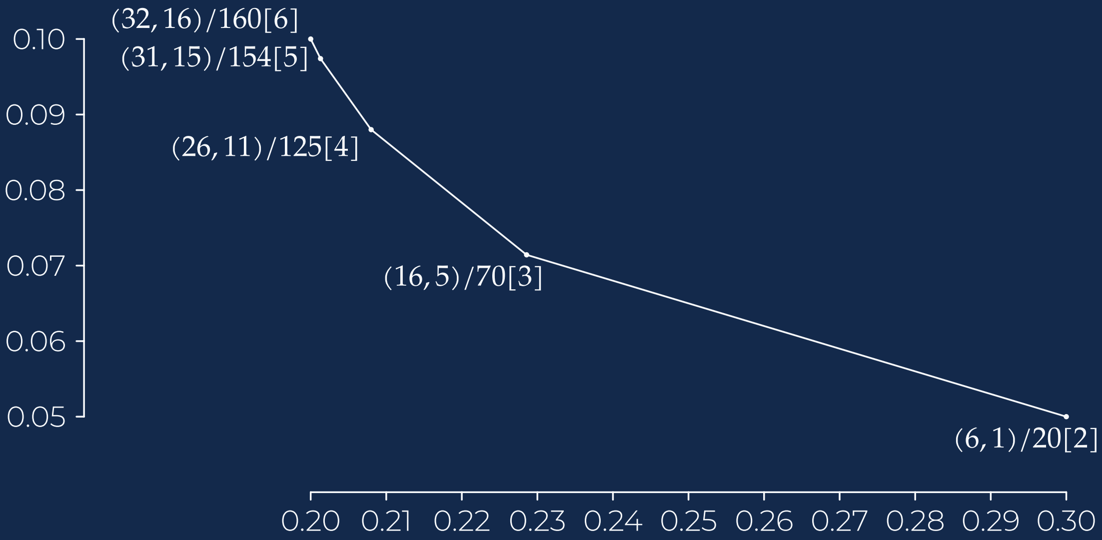
Some reductions for ease of comparison: (n, k, d) depends on actual applications.
So papers on this subject usually fix (n, k, d) and compare different codes' (α, β, M).

Researchers usually start from $n = d + 1$ and increase n later,
so it makes sense to ignore n and parametrize the comparison by (k, d).

It is also common to assume that the file is very very large, so
M can be pretty large and we only care about the long-term efficiency $(\alpha/M, \beta/M)$.

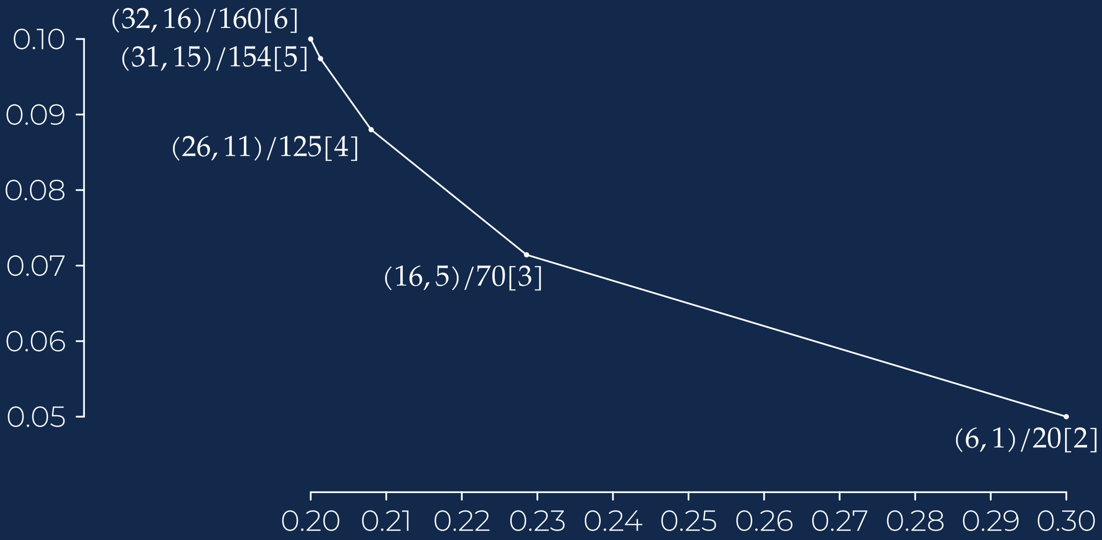
In short, it is reasonable to plot all possible $(\alpha/M, \beta/M)$ for each and every (k, d).

The performance of Moulin Code



Back to the opening example $(k, d) = (5, 6)$. These are the achievable $(\alpha/M, \beta/M)$.
The number in [square bracket] is the width of the diagram.

The performance of Moulin Code



Back to the opening example $(k, d) = (5, 6)$. These are the achievable $(\alpha/M, \beta/M)$.

The number in [square bracket] is the width of the diagram.

Open questions

Is Moulin Codes' construction isomorphic to any (well-)known structure?

Are Moulin Codes optimal in terms of the $(\alpha/M, \beta/M)$-plots?

Open questions

Is Moulin Codes' construction isomorphic to any (well-)known structure?

Are Moulin Codes optimal in terms of the $(\alpha/M, \beta/M)$-plots?

Thank you, Questions

Beamer available at https://ag21.symbol.codes (so the links below are included).

Paper version is titled *Multilinear Algebra for Distributed Storage*, is available at https://arxiv.org/abs/2006.08911, and is accepted for publication in SIAM SIAGA.

A similar work (that focuses on $M = k\alpha$ and uses symmetric algebra) is available at https://arxiv.org/abs/2006.16998 and accepted for publication in Springer AAECC.

Why is it called Moulin Code?

*Mu*ti*lin*ear algebra.

This code is inspired by another code called *Cascade Code*.

Cascade is also a type of waterfall.

Moulin is another type of waterfall, the one that appears in glaciers.

AWS Glacier is a popular storage service.