Part I

Linear codes

WISDOM

WISDOM

When you have eliminated impossible,

WISDOM

When you have eliminated impossible,

whatever remains,

WISDOM

When you have eliminated impossible,

whatever remains,
however impossible

When you have eliminated impossible,

whatever remains,
however impossible

must be

WISDOM

When you have eliminated impossible,

whatever remains,
however impossible

must be the TRUTH

EXERCISES

- For or solving the first set of exerises you have 3 weeks,

EXERCISES

- For or solving the first set of exerises you have 3 weeks,
- Seccond set of exercises is allready posted and solutions should be delivered not latr than October 29.

EXERCISES

- For or solving the first set of exerises you have 3 weeks,
- Seccond set of exercises is allready posted and solutions should be delivered not latr than October 29.
- All next sets of Exercises will be put on my web page and into IS, always at 18.00 on Thurdays after my lecture and solutions should be delvered in 2 weeks to you.

WHY LINEAR CODES?

Most of the important codes are special types of so-called linear codes.

WHY LINEAR CODES?

Most of the important codes are special types of so-called linear codes.

Linear codes are of very large importance because they have

WHY LINEAR CODES?

Most of the important codes are special types of so-called linear codes.

Linear codes are of very large importance because they have

very concise description,

WHY LINEAR CODES?

Most of the important codes are special types of so-called linear codes.

Linear codes are of very large importance because they have

very concise description,
very nice properties,

WHY LINEAR CODES?

Most of the important codes are special types of so-called linear codes.

Linear codes are of very large importance because they have

very concise description,
very nice properties,
very easy encoding

WHY LINEAR CODES?

Most of the important codes are special types of so-called linear codes.

Linear codes are of very large importance because they have

very concise description,
very nice properties,
very easy encoding

and, in general,

an easy to describe decoding.

WHY LINEAR CODES?

Most of the important codes are special types of so-called linear codes.

Linear codes are of very large importance because they have

very concise description,
very nice properties,
very easy encoding

and, in general,

an easy to describe decoding.

Many practically important linear codes have also an efficient decoding.

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\bmod q}$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{ mod } q}$ or $+_q$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\bmod q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\bmod q}$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\bmod q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\bmod q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 =$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\bmod q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\bmod q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{ mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{ mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 =$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 =$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\bmod q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\bmod q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 = 3$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 = 3 \quad 5 \times_7 5$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 = 3 \quad 5 \times_7 5 = 4$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{ mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{ mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 = 3 \quad 5 \times_7 5 = 4$

Example — $GF(11)$

$7 +_{11} 8 =$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{ mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{ mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 = 3 \quad 5 \times_7 5 = 4$

Example — $GF(11)$

$7 +_{11} 8 = 4$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 = 3 \quad 5 \times_7 5 = 4$

Example — $GF(11)$

$7 +_{11} 8 = 4 \quad 7 \times_{11} 8 =$

MATHEMATICS BEHIND - GALOIS FIELDS $GF(q)$ – with q a prime.

It is the set $\{0, 1, \ldots, q-1\}$ with two operations

addition modulo q — $+_{\text{mod } q}$ or $+_q$ or very simply $+$
multiplication modulo q — $\times_{\text{mod } q}$ or $\times_q$ or very simply $\times$ or $\cdot$

Example — $GF(3)$

$2 +_3 2 = 1 \quad 2 \times_3 2 = 1$

Example — $GF(7)$

$5 +_7 5 = 3 \quad 5 \times_7 5 = 4$

Example — $GF(11)$

$7 +_{11} 8 = 4 \quad 7 \times_{11} 8 = 1$

Comment. To design linear codes we will use Galois fields $GF(q)$ with q being a prime. One can also use Galois fields $GF(q^k)$, $k > 1$, but their structure and operations are defined in a more complex way, see the Appendix.

REPETITIONS - I.

REPETITIONS - I.

Given an alphabet Σ, any set $C \subset \Sigma^*$ is called a **code** and its elements are called **codewords**.

REPETITIONS - I.

Given an alphabet Σ, any set $C \subset \Sigma^*$ is called a **code** and its elements are called **codewords**.

By a **coding/encoding** of elements (messages) from a set M by codewords from a code C we understand any one-to-one mapping (encoder) e such that

$$e : M \to C$$

.

REPETITIONS - I.

Given an alphabet Σ, any set $C \subset \Sigma^*$ is called a **code** and its elements are called **codewords**.

By a **coding/encoding** of elements (messages) from a set M by codewords from a code C we understand any one-to-one mapping (encoder) e such that

$$e : M \to C$$

.

Encoding (code) is called systematic if for any $m \in M \subset \Sigma^*$

$$e(m) = mc_m \text{ for some } c_m \in \Sigma^*$$

SYSTEMATIC CODES I

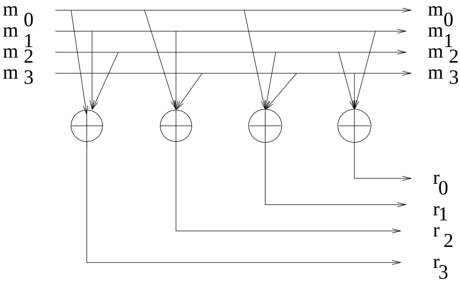
A code is called systematic if its encoder transmit a message (an input dataword) w into a codeword of the form wc_w, or (w, c_w). That is if the codeword for the message w consists of two parts: the message w itself (called also information part) and a redundancy part c_w

SYSTEMATIC CODES I

A code is called systematic if its encoder transmit a message (an input dataword) w into a codeword of the form wc_w, or (w, c_w). That is if the codeword for the message w consists of two parts: the message w itself (called also information part) and a redundancy part c_w

Nowadays most of the stream codes that are used in practice are systematic.

An example of a systematic encoder, that produces so called extended Hamming $(8, 4, 1)$ code is in the following figure.



1. A code C is said to be an (n, M, d) **code**, if

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.
   - Large M;

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.
   - Large M;
   - Large d;

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.
   - Large M;
   - Large d;
   - Encoding should be fast;

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.
   - Large M;
   - Large d;
   - Encoding should be fast; decoding reasonably efficient

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.
   - Large M;
   - Large d;
   - Encoding should be fast; decoding reasonably efficient
   - Encodings of similar messages should be very different.

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.
   - Large M;
   - Large d;
   - Encoding should be fast; decoding reasonably efficient
   - Encodings of similar messages should be very different.
   - Error corrections potential should be large.

REPETITIONS - II.

1. A code C is said to be an (n, M, d) **code**, if
   - n is the length of codewords in C
   - M is the number of codewords in C
   - d is the minimal distance of C
2. **A good code for encoding a set of messages should have**:
   - Small n.
   - Large M;
   - Large d;
   - Encoding should be fast; decoding reasonably efficient
   - Encodings of similar messages should be very different.
   - Error corrections potential should be large.

FRAMEWORK

FRAMEWORK

The goal of coding theory is to develop for a given set of **messgaes** M,

The goal of coding theory is to develop for a given set of **messgaes** M,

for example for the set of names of students/participants of this crypto lecture,

The goal of coding theory is to develop for a given set of **messgaes** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

FRAMEWORK

The goal of coding theory is to develop for a given set of **messgaes** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

for example **UČO**s

The goal of coding theory is to develop for a given set of **messgaes** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

for example **UČO**s

and to send through a noisy chanel UČO of students instead of their names,

FRAMEWORK

The goal of coding theory is to develop for a given set of **messgaes** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

for example **UČO**s

and to send through a noisy chanel UČO of students instead of their names,
in such a way that what will be received can be used to determine name that had to be transmitted

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČOs of any two student of this class will be again an UČO of some student of this class?

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČOs of any two student of this class will be again an UČO of some student of this class?

The answer is NO and the proof of that is almost trivial.

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČOs of any two student of this class will be again an UČO of some student of this class?

The answer is NO and the proof of that is almost trivial.

Is it possible to give to each student UČO in such a way that bit-wise sums of binary representations of UČOs of any two student of this class will be again binary representations of UČOs of some students of this class?

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČOs of any two student of this class will be again an UČO of some student of this class?

The answer is NO and the proof of that is almost trivial.

Is it possible to give to each student UČO in such a way that bit-wise sums of binary representations of UČOs of any two student of this class will be again binary representations of UČOs of some students of this class?

In general, does it has a sense to look for such codes that some important sum of any two codewords is again a codeword?

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n)$, $v = (v_1, v_2. \ldots, v_n)$ then
   $u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n))$

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n)$, $v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n)$)
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2. \ldots, v_n)$ then
   $u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n),$, then $au = (au_1, au_2, \ldots, au_n))$

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n)$, $v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n)$)
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n)$)

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
(if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
$u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n)$)
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n)$)

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n)$, $v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n)$)
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n)$)

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n, k]$-code.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n)$, $v = (v_1, v_2, \ldots, v_n)$ then
   $u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n)$)
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n)$)

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n, k]$-code. It has q^k codewords.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n),$, then $au = (au_1, au_2, \ldots, au_n))$

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n,k]$-code. It has q^k codewords. If the minimal distance of C is d, then it is said to be the $[n,k,d]$ code.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
(if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
$u + v = (u_1 +_q v_1, u_2 +_q v_2 \ldots, u_n +_q v_n)$)
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n)$)

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n, k]$-code. It has q^k codewords. If the minimal distance of C is d, then it is said to be the $[n, k, d]$ code.

LINEAR CODES - II.

If C is a linear $[n, k]$ code, then it has several bases.

LINEAR CODES - II.

If C is a linear $[n, k]$ code, then it has several bases.

A base **B** of C is such a sets of k codewods of C that each codeword of C is a linear combination of the codewords from the base **B**.

If C is a linear $[n, k]$ code, then it has several bases.

A base **B** of C is such a sets of k codewods of C that each codeword of C is a linear combination of the codewords from the base **B**.

Each base **B** of C is usually reperesented by a (k, n) matrix, $G_{\mathbf{B}}$, so called a **generator matrix of C**, the i-th row of which is the i-th codeword of **B**.

EXERCISE

EXERCISE

Which of the following binary codes are linear?

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

$C_3 = \{00000, 01101, 10110, 11011\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

$C_3 = \{00000, 01101, 10110, 11011\}$ – YES

$C_5 = \{101, 111, 011\}$ – NO

$C_6 = \{000, 001, 010, 011\}$ – YES

$C_7 = \{0000, 1001, 0110, 1110\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word,

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,
- all sums of two or more words in S.

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S\rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S\rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,
- all sums of two or more words in S.

Example

$$S = \{0100, 0011, 1100\}$$
$$\langle S\rangle = \{0000, 0100, 0011, 1100, 0111, 1011, 1000, 1111\}.$$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,
- all sums of two or more words in S.

Example

$$S = \{0100, 0011, 1100\}$$

$$\langle S \rangle = \{0000, 0100, 0011, 1100, 0111, 1011, 1000, 1111\}.$$

BASIC PROPERTIES of LINEAR CODES I

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x, y)$. Hence $h(C) = w(x - y) \geq w(C)$.

On the other hand, for some $x \in C$

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x, y)$. Hence $h(C) = w(x - y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x, 0) \geq h(C).$$

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x,y) = w(x-y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x,y)$. Hence $h(C) = w(x-y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x,0) \geq h(C).$$

Consequence

- If C is a non-linear code with m codewords, then in order to determine $h(C)$ one has to make in general $\binom{m}{2} = \Theta(m^2)$ comparisons in the worst case.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x, y)$. Hence $h(C) = w(x - y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x, 0) \geq h(C).$$

Consequence

- If C is a non-linear code with m codewords, then in order to determine $h(C)$ one has to make in general $\binom{m}{2} = \Theta(m^2)$ comparisons in the worst case.
- **If C is a linear code with m codewords, then in order to determine $h(C)$, $m - 1$ comparisons are enough.**

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x,y) = w(x-y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x,y)$. Hence $h(C) = w(x-y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x,0) \geq h(C).$$

Consequence

- If C is a non-linear code with m codewords, then in order to determine $h(C)$ one has to make in general $\binom{m}{2} = \Theta(m^2)$ comparisons in the worst case.
- **If C is a linear code with m codewords, then in order to determine $h(C)$, $m-1$ comparisons are enough.**

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n,k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

$$\{1111111, 1000101, 1100010, 0110001\}.$$

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

$$\{1111111, 1000101, 1100010, 0110001\}.$$

How many different bases has a linear code?

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

$$\{1111111, 1000101, 1100010, 0110001\}.$$

How many different bases has a linear code?

Theorem A binary linear code of dimension k has

$$\frac{1}{k!}\prod_{i=0}^{k-1}(2^k - 2^i)$$

bases.

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

However, In case we have $[2^{200}, 200]$ linear code C, then to specify/store fully C we need only to store

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

However, In case we have $[2^{200}, 200]$ linear code C, then to specify/store fully C we need only to store

200

codewords

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

However, In case we have $[2^{200}, 200]$ linear code C, then to specify/store fully C we need only to store

200

codewords - from one of its basis.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

Definition A $k \times n$ matrix whose rows form a basis of a linear $[n, k]$-code (subspace) C is said to be the generator matrix of C.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

Definition A $k \times n$ matrix whose rows form a basis of a linear $[n, k]$-code (subspace) C is said to be the generator matrix of C.

Example One of the generator matrices of the binary code

$$C_2 = \left\{\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{matrix}\right\} \text{ is the matrix } \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

Definition A $k \times n$ matrix whose rows form a basis of a linear $[n, k]$-code (subspace) C is said to be the generator matrix of C.

Example One of the generator matrices of the binary code

$$C_2 = \left\{\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{matrix}\right\} \text{ is the matrix } \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

and one of the generator matrices of the code

$$C_4 \text{ is } \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

Definition A $k \times n$ matrix whose rows form a basis of a linear $[n, k]$-code (subspace) C is said to be the generator matrix of C.

Example One of the generator matrices of the binary code

$$C_2 = \left\{\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{matrix}\right\} \text{ is the matrix } \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

and one of the generator matrices of the code

$$C_4 \text{ is } \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

3. There are simple encoding/decoding procedures for linear codes.

ADANTAGES and DISADVANTAGES of LINEAR CODES II.

Disadvantages of linear codes are small:

1. Linear q-codes are not defined unless q is a power of a prime.

Disadvantages of linear codes are small:

1. Linear q-codes are not defined unless q is a power of a prime.
2. The restriction to linear codes might be a restriction to weaker codes than sometimes desired.

EQUIVALENCE of LINEAR CODES I

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a permutation of the words or positions of the code;

(b multiplication of symbols appearing in a fixed position by a non-zero scalar.

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

- (a permutation of the words or positions of the code;
- (b multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

(c) addition of one row to another

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

(c) addition of one row to another

(d) permutation of columns

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;
(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows
(b) multiplication of a row by a non-zero scalar
(c) addition of one row to another
(d) permutation of columns
(e) multiplication of a column by a non-zero scalar

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

(c) addition of one row to another

(d) permutation of columns

(e) multiplication of a column by a non-zero scalar

Proof Operations (a) - (c) just replace one basis by another. Last two operations convert a generator matrix to one of an equivalent code.

EQUIVALENCE of LINEAR CODES II

Theorem Let G be a generator matrix of an $[n, k]$-code. Rows of G are then linearly independent .By operations (a) - (e) the matrix G can be transformed into the form: $[I_k|A]$ where I_k is the $k \times k$ identity matrix, and A is a $k \times (n-k)$ matrix.

EQUIVALENCE of LINEAR CODES II

Theorem Let G be a generator matrix of an $[n, k]$-code. Rows of G are then linearly independent .By operations (a) - (e) the matrix G can be transformed into the form: $[I_k|A]$ where I_k is the $k \times k$ identity matrix, and A is a $k \times (n-k)$ matrix.

Example

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \rightarrow$$

$$\rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \rightarrow$$

ENCODING with LINEAR CODES

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

ENCODING with LINEAR CODES

is a vector × matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

ENCODING with LINEAR CODES

is a vector × matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n,k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

ENCODING with LINEAR CODES

is a vector × matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \dots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \dots, r_k \text{ are rows of } G.$$

Example Let C be a $[7, 4]$-code with the generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7, 4]$-code with the generator matrix

$$G= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? .....

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7, 4]$-code with the generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? 0000000
1 0 0 0 is encoded as? .....

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n,k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7,4]$-code with the generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? 0000000
1 0 0 0 is encoded as? 1000101
1 1 1 0 is encoded as? .....

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7, 4]$-code with the generator matrix

$$\text{G}= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? 0000000
1 0 0 0 is encoded as? 1000101
1 1 1 0 is encoded as? 1110100

UNIQUENESS of ENCODING

with linear codes

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^k$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = uG$$

ranges over all 2^k codewords of C as u ranges over all 2^k messages of length k.

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^{k}$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = uG$$

ranges over all 2^k codewords of C as u ranges over all 2^k messages of length k. Therefore,

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^{k}$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = uG$$

ranges over all 2^k codewords of C as u ranges over all 2^k messages of length k. Therefore,

$$C = \{uG \mid u \in \{0,1\}^k\}$$

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^k$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = uG$$

ranges over all 2^k codewords of C as u ranges over all 2^k messages of length k. Therefore,

$$C = \{uG \mid u \in \{0,1\}^k\}$$

Moreover,

$$u_1 G = u_2 G$$

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^{k}$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = uG$$

ranges over all 2^k codewords of C as u ranges over all 2^k messages of length k. Therefore,

$$C = \{uG \mid u \in \{0,1\}^k\}$$

Moreover,

$$u_1 G = u_2 G$$

if and only if

$$u_1 = u_2.$$

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^k$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = uG$$

ranges over all 2^k codewords of C as u ranges over all 2^k messages of length k. Therefore,

$$C = \{uG \mid u \in \{0,1\}^k\}$$

Moreover,

$$u_1 G = u_2 G$$

if and only if

$$u_1 = u_2.$$

Proof If $u_1 G - u_2 G = 0$, then

$$0 = \sum_{i=1}^k u_{1,i} w_i - \sum_{i=1}^k u_{2,i} w_i = \sum_{i=1}^k (u_{1,i} - u_{2,i}) w_i$$

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^k$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = uG$$

ranges over all 2^k codewords of C as u ranges over all 2^k messages of length k. Therefore,

$$C = \{uG \mid u \in \{0,1\}^k\}$$

Moreover,

$$u_1 G = u_2 G$$

if and only if

$$u_1 = u_2.$$

Proof If $u_1 G - u_2 G = 0$, then

$$0 = \sum_{i=1}^k u_{1,i} w_i - \sum_{i=1}^k u_{2,i} w_i = \sum_{i=1}^k (u_{1,i} - u_{2,i}) w_i$$

And, therefore, since w_i are linearly independent, $u_1 = u_2$.

LINEAR CODES as SYSTEMATIC CODES

LINEAR CODES as SYSTEMATIC CODES

Since to each linear $[n, k]$-code C there is a generator matrix of the form $G = [I_k|A]$ an encoding of a dataword w with G has the form

$$wG = w \cdot wA$$

Each linear code is therefore equivalent to a systematic code.

DECODING of LINEAR CODES - BASICS

Decoding problem:

DECODING of LINEAR CODES - BASICS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent

DECODING of LINEAR CODES - BASICS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent

and the word $y = y_1 \dots y_n$ is received,

DECODING of LINEAR CODES - BASICS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent

and the word $y = y_1 \dots y_n$ is received,

then $e = y - x = e_1 \dots e_n$ is said to be the error vector.

DECODING of LINEAR CODES - BASICS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent

and the word $y = y_1 \dots y_n$ is received,

then $e = y - x = e_1 \dots e_n$ is said to be the error vector.

The decoder must therefore decide, given y,

which x was sent,

DECODING of LINEAR CODES - BASICS

Decoding problem: If a codeword: $x = x_1 \ldots x_n$ is sent

and the word $y = y_1 \ldots y_n$ is received,

then $e = y - x = e_1 \ldots e_n$ is said to be the error vector.

The decoder must therefore decide, given y,

which x was sent,

or, equivalently, which error e occurred.

DECODING of LINEAR CODES - METHOD of COSETS

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**.

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** **(u-coset) of C in F_q^n.**

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \ldots x_n$ is sent and the word $y = y_1 \ldots y_n$ is received, then $e = y - x = e_1 \ldots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \mid x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \ldots x_n$ is sent and the word $y = y_1 \ldots y_n$ is received, then $e = y - x = e_1 \ldots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C =$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \ldots x_n$ is sent and the word $y = y_1 \ldots y_n$ is received, then $e = y - x = e_1 \ldots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \mid x \in C\}$$

is called a **coset** (**u-coset**) **of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$
Cosets:
$0000 + C = C$,

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C$,

$1000 + C =$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset**) **of** C **in** F_q^n**.**

Example Let $C = \{0000, 1011, 0101, 1110\}$
Cosets:
$0000 + C = C$,
$1000 + C = \{1000$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$
Cosets:
$0000 + C = C$,
$1000 + C = \{1000, 0011,$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C$,

$1000 + C = \{1000, 0011, 1101,$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$
Cosets:
$0000 + C = C$,
$1000 + C = \{1000, 0011, 1101, 0110\}$,

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \mid x \in C\}$$

is called a **coset** (**u-coset**) **of** C **in** F_q^n.

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C,$

$1000 + C = \{1000, 0011, 1101, 0110\},$

$0100 + C =$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$
Cosets:
$0000 + C = C,$
$1000 + C = \{1000, 0011, 1101, 0110\},$
$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C,$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \ldots x_n$ is sent and the word $y = y_1 \ldots y_n$ is received, then $e = y - x = e_1 \ldots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset**) **of** C **in** F_q^n.

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C,$

$1000 + C = \{1000, 0011, 1101, 0110\},$

$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C,$

$0010 + C =$

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset**) **of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C$,
$1000 + C = \{1000, 0011, 1101, 0110\}$,
$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C$,
$0010 + C = \{0010, 1001, 0111, 1100\}$.

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \mid x \in C\}$$

is called a **coset** (**u-coset**) **of** C **in** F_q^n.

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C,$

$1000 + C = \{1000, 0011, 1101, 0110\},$

$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C,$

$0010 + C = \{0010, 1001, 0111, 1100\}.$

Are there some other cosets in this case?

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (**u-coset**) **of** C **in** F_q^n.

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C,$

$1000 + C = \{1000, 0011, 1101, 0110\},$

$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C,$

$0010 + C = \{0010, 1001, 0111, 1100\}.$

Are there some other cosets in this case?

Theorem Suppose C is a linear $[n, k]$-code over F_q^n. Then

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** (u-**coset**) **of** C **in** F_q^n.

Example Let $C = \{0000, 1011, 0101, 1110\}$
Cosets:
$0000 + C = C$,
$1000 + C = \{1000, 0011, 1101, 0110\}$,
$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C$,
$0010 + C = \{0010, 1001, 0111, 1100\}$.
Are there some other cosets in this case?

Theorem Suppose C is a linear $[n, k]$-code over F_q^n. Then

(a) every vector of F_q^n is in some coset of C,

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \,|\, x \in C\}$$

is called a **coset** **(u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C$,

$1000 + C = \{1000, 0011, 1101, 0110\}$,

$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C$,

$0010 + C = \{0010, 1001, 0111, 1100\}$.

Are there some other cosets in this case?

Theorem Suppose C is a linear $[n, k]$-code over F_q^n. Then

(a every vector of F_q^n is in some coset of C,

(b every coset contains exactly q^k elements,

DECODING of LINEAR CODES - METHOD of COSETS

Decoding problem: If a codeword: $x = x_1 \dots x_n$ is sent and the word $y = y_1 \dots y_n$ is received, then $e = y - x = e_1 \dots e_n$ is said to be the **error vector**. The decoder must decide, from y, which x was sent, or, equivalently, which error e occurred.

To describe so called **Decoding cosets method** the concept of **cosets** has to be introduced:

Definition Suppose C is an $[n, k]$-code over F_q^n and $u \in F_q^n$. Then the set

$$u + C = \{u + x \mid x \in C\}$$

is called a **coset** (**u-coset) of C in F_q^n.**

Example Let $C = \{0000, 1011, 0101, 1110\}$

Cosets:

$0000 + C = C$,

$1000 + C = \{1000, 0011, 1101, 0110\}$,

$0100 + C = \{0100, 1111, 0001, 1010\} = 0001 + C$,

$0010 + C = \{0010, 1001, 0111, 1100\}$.

Are there some other cosets in this case?

Theorem Suppose C is a linear $[n, k]$-code over F_q^n. Then

a. every vector of F_q^n is in some coset of C,
b. every coset contains exactly q^k elements,
c. two cosets are either disjoint or identical.

NEAREST NEIGHBOUR DECODING SCHEME

NEAREST NEIGHBOUR DECODING SCHEME

Each vector having minimum weight in a coset is called a **coset leader**.

NEAREST NEIGHBOUR DECODING SCHEME

Each vector having minimum weight in a coset is called a **coset leader**.

1. Design a **(Slepian) standard array** for an $[n, k]$-code C - that is a $q^{n-k} \times q^k$ array of the form:

NEAREST NEIGHBOUR DECODING SCHEME

Each vector having minimum weight in a coset is called a **coset leader**.

1. Design a **(Slepian) standard array** for an $[n, k]$-code C - that is a $q^{n-k} \times q^k$ array of the form:

codewords	coset leader	codeword 2	...	codeword 2^k
	coset leader	$+$	...	$+$
	...	$+$	$+$	$+$
	coset leader	$+$	...	$+$
	coset leader			

where codewords of C are in the first row and elements of each coset are in a special row, with some of the cosets leaders in the frst column.

NEAREST NEIGHBOUR DECODING SCHEME

Each vector having minimum weight in a coset is called a **coset leader**.

1. Design a **(Slepian) standard array** for an $[n, k]$-code C - that is a $q^{n-k} \times q^k$ array of the form:

codewords	coset leader	codeword 2	...	codeword 2^k
	coset leader	$+$	...	$+$
	...	$+$	$+$	$+$
	coset leader	$+$	...	$+$
	coset leader			

where codewords of C are in the first row and elements of each coset are in a special row, with some of the cosets leaders in the frst column.

Example

0000	1011	0101	1110
1000	0011	1101	0110
0100	1111	0001	1010
0010	1001	0111	1100

NEAREST NEIGHBOUR DECODING SCHEME

Each vector having minimum weight in a coset is called a **coset leader**.

1. Design a **(Slepian) standard array** for an $[n, k]$-code C - that is a $q^{n-k} \times q^k$ array of the form:

codewords	coset leader	codeword 2	...	codeword 2^k
	coset leader	+	...	+
	...	+	+	+
	coset leader	+	...	+
	coset leader			

where codewords of C are in the first row and elements of each coset are in a special row, with some of the cosets leaders in the frst column.

Example

0000	1011	0101	1110
1000	0011	1101	0110
0100	1111	0001	1010
0010	1001	0111	1100

A received word y is decoded as the codeword in the first row of the column in which y occurs.

NEAREST NEIGHBOUR DECODING SCHEME

Each vector having minimum weight in a coset is called a **coset leader**.

1. Design a **(Slepian) standard array** for an $[n, k]$-code C - that is a $q^{n-k} \times q^k$ array of the form:

codewords	coset leader	codeword 2	...	codeword 2^k
	coset leader	+	...	+
	...	+	+	+
	coset leader	+	...	+
	coset leader			

where codewords of C are in the first row and elements of each coset are in a special row, with some of the cosets leaders in the frst column.

Example

0000	1011	0101	1110
1000	0011	1101	0110
0100	1111	0001	1010
0010	1001	0111	1100

A received word y is decoded as the codeword in the first row of the column in which y occurs.

Error vectors which will be corrected are precisely coset leaders!

NEAREST NEIGHBOUR DECODING SCHEME

Each vector having minimum weight in a coset is called a **coset leader**.

1. Design a **(Slepian) standard array** for an $[n, k]$-code C - that is a $q^{n-k} \times q^k$ array of the form:

codewords	coset leader	codeword 2	...	codeword 2^k
	coset leader	+	...	+
	...	+	+	+
	coset leader	+	...	+
	coset leader			

where codewords of C are in the first row and elements of each coset are in a special row, with some of the cosets leaders in the frst column.

Example

0000	1011	0101	1110
1000	0011	1101	0110
0100	1111	0001	1010
0010	1001	0111	1100

A received word y is decoded as the codeword in the first row of the column in which y occurs.

Error vectors which will be corrected are precisely coset leaders!

In practice, this decoding method is too slow and requires too much memory.

How good are particular linear codes?

PROBABILITY of GOOD ERROR CORRECTION

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

Probability of an error in the case of a given error vector of weight i is

$$p^i(1-p)^{n-i}.$$

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

Probability of an error in the case of a given error vector of weight i is

$$p^i(1-p)^{n-i}.$$

Therefore, it holds.

Theorem Let C be a binary $[n,k]$-code, and for $i = 0, 1, \ldots, n$ let α_i be the number of coset leaders of weight i.

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

Probability of an error in the case of a given error vector of weight i is

$$p^i(1-p)^{n-i}.$$

Therefore, it holds.

Theorem Let C be a binary $[n,k]$-code, and for $i = 0, 1, \ldots, n$ let α_i be the number of coset leaders of weight i. The probability $P_{corr}(C)$ that a received vector, when decoded by means of a standard array, is the codeword which was sent is given by

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

Probability of an error in the case of a given error vector of weight i is

$$p^i(1-p)^{n-i}.$$

Therefore, it holds.

Theorem Let C be a binary $[n, k]$-code, and for $i = 0, 1, \ldots, n$ let α_i be the number of coset leaders of weight i. The probability $P_{corr}(C)$ that a received vector, when decoded by means of a standard array, is the codeword which was sent is given by

$$P_{corr}(C) = \sum_{i=0}^{n} \alpha_i p^i(1-p)^{n-i}.$$

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

Probability of an error in the case of a given error vector of weight i is

$$p^i(1-p)^{n-i}.$$

Therefore, it holds.

Theorem Let C be a binary $[n,k]$-code, and for $i = 0, 1, \ldots, n$ let α_i be the number of coset leaders of weight i. The probability $P_{corr}(C)$ that a received vector, when decoded by means of a standard array, is the codeword which was sent is given by

$$P_{corr}(C) = \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}.$$

Example For the $[4,2]$-code of the last example

$$\alpha_0 = 1, \alpha_1 = 3, \alpha_2 = \alpha_3 = \alpha_4 = 0.$$

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

Probability of an error in the case of a given error vector of weight i is

$$p^i(1-p)^{n-i}.$$

Therefore, it holds.

Theorem Let C be a binary $[n,k]$-code, and for $i = 0,1,\dots,n$ let α_i be the number of coset leaders of weight i. The probability $P_{corr}(C)$ that a received vector, when decoded by means of a standard array, is the codeword which was sent is given by

$$P_{corr}(C) = \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}.$$

Example For the $[4,2]$-code of the last example

$$\alpha_0 = 1, \alpha_1 = 3, \alpha_2 = \alpha_3 = \alpha_4 = 0.$$

Hence

$$P_{corr}(C) = (1-p)^4 + 3p(1-p)^3 = (1-p)^3(1+2p).$$

PROBABILITY of GOOD ERROR CORRECTION

What is the probability that a received word will be decoded correctly -that is as the codeword that was sent (for binary linear codes and binary symmetric channel)?

Probability of an error in the case of a given error vector of weight i is

$$p^i(1-p)^{n-i}.$$

Therefore, it holds.

Theorem Let C be a binary $[n, k]$-code, and for $i = 0, 1, \ldots, n$ let α_i be the number of coset leaders of weight i. The probability $P_{corr}(C)$ that a received vector, when decoded by means of a standard array, is the codeword which was sent is given by

$$P_{corr}(C) = \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}.$$

Example For the $[4, 2]$-code of the last example

$$\alpha_0 = 1, \alpha_1 = 3, \alpha_2 = \alpha_3 = \alpha_4 = 0.$$

Hence

$$P_{corr}(C) = (1-p)^4 + 3p(1-p)^3 = (1-p)^3(1+2p).$$

If $p = 0.01$, then $P_{corr} = 0.9897$

PROBABILITY of GOOD ERROR DETECTION

PROBABILITY of GOOD ERROR DETECTION

Suppose a binary linear code is used only for error detection.

PROBABILITY of GOOD ERROR DETECTION

Suppose a binary linear code is used only for error detection.

The decoder will fail to detect errors which have occurred if the received word y is a codeword different from the codeword x which was sent, i. e. if the error vector $e = y - x$ is itself a non-zero codeword.

PROBABILITY of GOOD ERROR DETECTION

Suppose a binary linear code is used only for error detection.

The decoder will fail to detect errors which have occurred if the received word y is a codeword different from the codeword x which was sent, i. e. if the error vector $e = y - x$ is itself a non-zero codeword.

The probability $P_{undetect}(C)$ that an incorrect codeword is received is given by the following result.

PROBABILITY of GOOD ERROR DETECTION

Suppose a binary linear code is used only for error detection.

The decoder will fail to detect errors which have occurred if the received word y is a codeword different from the codeword x which was sent, i. e. if the error vector $e = y - x$ is itself a non-zero codeword.

The probability $P_{undetect}(C)$ that an incorrect codeword is received is given by the following result.

Theorem Let C be a binary $[n, k]$-code and let A_i denote the number of codewords of C of weight i. Then, if C is used for error detection, the probability of an incorrect message being received is

$$P_{undetect}(C) = \sum_{i=0}^{n} A_i p^i (1-p)^{n-i}.$$

PROBABILITY of GOOD ERROR DETECTION

Suppose a binary linear code is used only for error detection.

The decoder will fail to detect errors which have occurred if the received word y is a codeword different from the codeword x which was sent, i. e. if the error vector $e = y - x$ is itself a non-zero codeword.

The probability $P_{undetect}(C)$ that an incorrect codeword is received is given by the following result.

Theorem Let C be a binary $[n, k]$-code and let A_i denote the number of codewords of C of weight i. Then, if C is used for error detection, the probability of an incorrect message being received is

$$P_{undetect}(C) = \sum_{i=0}^{n} A_i p^i (1-p)^{n-i}.$$

Example In the case of the $[4, 2]$ code from the last example

$$A_2 = 1 \; A_3 = 2$$
$$P_{undetect}(C) = p^2(1-p)^2 + 2p^3(1-p) = p^2 - p^4.$$

PROBABILITY of GOOD ERROR DETECTION

Suppose a binary linear code is used only for error detection.

The decoder will fail to detect errors which have occurred if the received word y is a codeword different from the codeword x which was sent, i. e. if the error vector $e = y - x$ is itself a non-zero codeword.

The probability $P_{undetect}(C)$ that an incorrect codeword is received is given by the following result.

Theorem Let C be a binary $[n, k]$-code and let A_i denote the number of codewords of C of weight i. Then, if C is used for error detection, the probability of an incorrect message being received is

$$P_{undetect}(C) = \sum_{i=0}^{n} A_i p^i (1-p)^{n-i}.$$

Example In the case of the $[4, 2]$ code from the last example

$$A_2 = 1 \; A_3 = 2$$
$$P_{undetect}(C) = p^2(1-p)^2 + 2p^3(1-p) = p^2 - p^4.$$

For $p = 0.01$

$$P_{undetect}(C) = 0.00009999.$$

SYNDROMES APPROACH to DECODING

DUAL CODES

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \ldots u_n, \quad v = v_1 \ldots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \ldots + u_n v_n.$$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 =$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \ldots u_n, \quad v = v_1 \ldots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \ldots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 =$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 = 2$
$1212 \cdot 2121 =$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example $In \quad F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 = 2$
$1212 \cdot 2121 = 2$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 = 2$
$1212 \cdot 2121 = 2$

If $u \cdot v = 0$ then words (vectors) u and v are called **orthogonal words**.

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example $In \quad F_2^4 : 1001 \cdot 1001 = 0$
In $\quad F_3^4 : 2001 \cdot 1210 = 2$
$\qquad 1212 \cdot 2121 = 2$

If $u \cdot v = 0$ then words (vectors) u and v are called **orthogonal words**.

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 = 2$
$1212 \cdot 2121 = 2$

If $u \cdot v = 0$ then words (vectors) u and v are called **orthogonal words**.

Given a linear $[n, k]$-code C, then the dual code of C, denoted by $C^{\perp}$, is defined by

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 = 2$
$1212 \cdot 2121 = 2$

If $u \cdot v = 0$ then words (vectors) u and v are called **orthogonal words**.

Given a linear $[n, k]$-code C, then the dual code of C, denoted by $C^\perp$, is defined by

$$C^\perp = \{v \in F_q^n \mid v \cdot u = 0 \text{ for all } u \in C\}.$$

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \ldots u_n, \quad v = v_1 \ldots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \ldots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 = 2$
$1212 \cdot 2121 = 2$

If $u \cdot v = 0$ then words (vectors) u and v are called **orthogonal words**.

Given a linear $[n, k]$-code C, then the dual code of C, denoted by $C^{\perp}$, is defined by

$$C^{\perp} = \{v \in F_q^n \mid v \cdot u = 0 \text{ for all } u \in C\}.$$

Lemma Suppose C is an $[n, k]$-code having a generator matrix G. Then for $v \in F_q^n$

$$v \in C^{\perp} \Leftrightarrow vG^{\top} = 0,$$

where $G^{\top}$ denotes the transpose of the matrix G.

DUAL CODES

Inner product of two vectors (words)

$$u = u_1 \dots u_n, \quad v = v_1 \dots v_n$$

in F_q^n is an element of $GF(q)$ defined (using modulo q operations) by

$$u \cdot v = u_1 v_1 + \dots + u_n v_n.$$

Example *In* $F_2^4 : 1001 \cdot 1001 = 0$
In $F_3^4 : 2001 \cdot 1210 = 2$
$1212 \cdot 2121 = 2$

If $u \cdot v = 0$ then words (vectors) u and v are called **orthogonal words**.

Given a linear $[n, k]$-code C, then the dual code of C, denoted by $C^\perp$, is defined by

$$C^\perp = \{v \in F_q^n \mid v \cdot u = 0 \text{ for all } u \in C\}.$$

Lemma Suppose C is an $[n, k]$-code having a generator matrix G. Then for $v \in F_q^n$

$$v \in C^\perp \Leftrightarrow vG^\top = 0,$$

where $G^\top$ denotes the transpose of the matrix G.

PARITY CHECK MATRICES

Each $[n, n-k]$ generator matrix H of an dual code $C^\perp$ of an $[n, k]$ liner code C is said to be a **parity check matrix for the code** C,

PARITE CHECKS versus ORTHOGONALITY

For understanding of the role the parity checks play for linear codes, it is important to understand the relation between orthogonality and general parity checks.

PARITE CHECKS versus ORTHOGONALITY

For understanding of the role the parity checks play for linear codes, it is important to understand the relation between orthogonality and general parity checks.

If binary words x and y are orthogonal, then the word y has even number of ones (1's) in the positions determined by ones (1's) in the word x.

PARITE CHECKS versus ORTHOGONALITY

For understanding of the role the parity checks play for linear codes, it is important to understand the relation between orthogonality and general parity checks.

If binary words x and y are orthogonal, then the word y has even number of ones (1's) in the positions determined by ones (1's) in the word x.

This implies that if words x and y are orthogonal, then x is a parity check word for y and y is a parity check word for x.

PARITE CHECKS versus ORTHOGONALITY

For understanding of the role the parity checks play for linear codes, it is important to understand the relation between orthogonality and general parity checks.

If binary words x and y are orthogonal, then the word y has even number of ones (1's) in the positions determined by ones (1's) in the word x.

This implies that if words x and y are orthogonal, then x is a parity check word for y and y is a parity check word for x.

Exercise: Let the word

$$100001$$

be orthogonal to all words of a set S of binary words of length 6.

PARITE CHECKS versus ORTHOGONALITY

For understanding of the role the parity checks play for linear codes, it is important to understand the relation between orthogonality and general parity checks.

If binary words x and y are orthogonal, then the word y has even number of ones (1's) in the positions determined by ones (1's) in the word x.

This implies that if words x and y are orthogonal, then x is a parity check word for y and y is a parity check word for x.

Exercise: Let the word

$$100001$$

be orthogonal to all words of a set S of binary words of length 6. What can we say about the words in S?

PARITE CHECKS versus ORTHOGONALITY

For understanding of the role the parity checks play for linear codes, it is important to understand the relation between orthogonality and general parity checks.

If binary words x and y are orthogonal, then the word y has even number of ones (1's) in the positions determined by ones (1's) in the word x.

This implies that if words x and y are orthogonal, then x is a parity check word for y and y is a parity check word for x.

Exercise: Let the word

$$100001$$

be orthogonal to all words of a set S of binary words of length 6. What can we say about the words in S?

Answer: All words of S have at the end the same symbol as at the beginning.

EXAMPLE

For the $[n, 1]$-repetition (binary) code C, with the generator matrix

$$G = (1, 1, \ldots, 1)$$

the dual code $C^\perp$ is $[n, n-1]$-code with the generator matrix $G^\perp$, described by

$$G^\perp = \begin{pmatrix} 1 & 1 & 0 & 0 & \ldots & 0 \\ 1 & 0 & 1 & 0 & \ldots & 0 \\ & \ldots & & & & \\ 1 & 0 & 0 & 0 & \ldots & 1 \end{pmatrix}$$

PARITY CHECK MATRICES I

Example If

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix},$$

PARITY CHECK MATRICES I

Example If

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}, \text{ then } C_5^{\perp} = C_5.$$

PARITY CHECK MATRICES I

Example If

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}, \text{ then } C_5^{\perp} = C_5.$$

If

$$C_6 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \text{ then}$$

PARITY CHECK MATRICES I

Example If

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}, \text{ then } C_5^{\perp} = C_5.$$

If

$$C_6 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \text{ then } C_6^{\perp} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}.$$

PARITY CHECK MATRICES I

Example If

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}, \text{ then } C_5^{\perp} = C_5.$$

If

$$C_6 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \text{ then } C_6^{\perp} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}.$$

Theorem Suppose C is a linear $[n, k]$-code over F_q^n, then the dual code $C^{\perp}$ is a linear $[n, n-k]$-code.

PARITY CHECK MATRICES I

Example If

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}, \text{ then } C_5^{\perp} = C_5.$$

If

$$C_6 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, \text{ then } C_6^{\perp} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}.$$

Theorem Suppose C is a linear $[n, k]$-code over F_q^n, then the dual code $C^{\perp}$ is a linear $[n, n-k]$-code.

Definition A parity-check matrix H for an $[n, k]$-code C is any generator matrix of $C^{\perp}$.

PARITY CHECK MATRICES

Definition A parity-check matrix H for an $[n,k]$-code C is any generator matrix of $C^{\perp}$.

PARITY CHECK MATRICES

Definition A parity-check matrix H for an $[n, k]$-code C is any generator matrix of $C^{\perp}$.

Theorem If H is a parity-check matrix of C, then

$$C = \{x \in F_q^n \,|\, xH^{\top} = 0\},$$

and therefore any linear code is completely specified by a parity-check matrix.

PARITY CHECK MATRICES

Definition A parity-check matrix H for an $[n, k]$-code C is any generator matrix of $C^{\perp}$.

Theorem If H is a parity-check matrix of C, then

$$C = \{x \in F_q^n \,|\, xH^{\top} = 0\},$$

and therefore any linear code is completely specified by a parity-check matrix.

Example Parity-check matrix for

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \text{ is } \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

PARITY CHECK MATRICES

Definition A parity-check matrix H for an $[n, k]$-code C is any generator matrix of $C^{\perp}$.

Theorem If H is a parity-check matrix of C, then

$$C = \{x \in F_q^n \mid xH^{\top} = 0\},$$

and therefore any linear code is completely specified by a parity-check matrix.

Example Parity-check matrix for

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \text{ is } \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

and for

$$C_6 \text{ is } \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

PARITY CHECK MATRICES

Definition A parity-check matrix H for an $[n,k]$-code C is any generator matrix of $C^{\perp}$.

Theorem If H is a parity-check matrix of C, then

$$C = \{x \in F_q^n \mid xH^{\top} = 0\},$$

and therefore any linear code is completely specified by a parity-check matrix.

Example Parity-check matrix for

$$C_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \text{ is } \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

and for

$$C_6 \text{ is } \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

The rows of a parity check matrix are parity checks on codewords. They actually say that certain linear combinations of elements of every codeword are zeros modulo 2.

SYNDROME DECODING

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n,k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

SYNDROME DECODING

Theorem If $G = [I_k | A]$ is the standard form generator matrix of an $[n, k]$-code C, then a parity check matrix for C is $H = [A^\top | I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left| I_4 \begin{vmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix} \right| \Rightarrow \text{parity check m. } H = \left| \begin{vmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{vmatrix} I_3 \right|$$

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n,k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left|\, I_4 \begin{vmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix} \right| \Rightarrow \text{parity check m. } H = \left| \begin{vmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{vmatrix} I_3 \right|$$

Definition Suppose H is a parity-check matrix of an $[n,k]$-code C. Then for any $y \in F_q^n$ the following word is called the syndrome of y:

$$S(y) = yH^\top.$$

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n, k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left| I_4 \begin{vmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix} \right| \Rightarrow \text{parity check m. } H = \left| \begin{vmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{vmatrix} I_3 \right|$$

Definition Suppose H is a parity-check matrix of an $[n, k]$-code C. Then for any $y \in F_q^n$ the following word is called the syndrome of y:

$$S(y) = yH^\top.$$

Lemma Two words have the same syndrome iff they are in the same coset.

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n, k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left|I_4 \left|\begin{matrix}1 & 0 & 1\\ 1 & 1 & 1\\ 1 & 1 & 0\\ 0 & 1 & 1\end{matrix}\right|\right| \Rightarrow \text{parity check m. } H = \left|\left|\begin{matrix}1 & 1 & 1 & 0\\ 0 & 1 & 1 & 1\\ 1 & 1 & 0 & 1\end{matrix}\right| I_3\right|$$

Definition Suppose H is a parity-check matrix of an $[n, k]$-code C. Then for any $y \in F_q^n$ the following word is called the syndrome of y:

$$S(y) = yH^\top.$$

Lemma Two words have the same syndrome iff they are in the same coset.

Syndrom decoding Assume that a standard array of an $[n, k]$ code C is given and, in addition, let in the last $n - k$ columns the syndrome for each coset be given.

$$\begin{array}{cccc|cccc|cccc|cccc|cc}
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0\\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1\\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1\\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0
\end{array}$$

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n,k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left| I_4 \left| \begin{matrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{matrix} \right| \right| \Rightarrow \text{parity check m. } H = \left| \left| \begin{matrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{matrix} \right| I_3 \right|$$

Definition Suppose H is a parity-check matrix of an $[n,k]$-code C. Then for any $y \in F_q^n$ the following word is called the syndrome of y:

$$S(y) = yH^\top.$$

Lemma Two words have the same syndrome iff they are in the same coset.

Syndrom decoding Assume that a standard array of an $[n,k]$ code C is given and, in addition, let in the last $n-k$ columns the syndrome for each coset be given.

$$\begin{array}{cccc|cccc|cccc|cccc|cc} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \end{array}$$

When a word y is received, then compute $S(y) = yH^\top$,

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n, k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left|\, I_4 \left|\begin{matrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{matrix}\right|\right. \Rightarrow \text{parity check m. } H = \left|\begin{matrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{matrix}\right| I_3 \,\Bigg|$$

Definition Suppose H is a parity-check matrix of an $[n, k]$-code C. Then for any $y \in F_q^n$ the following word is called the syndrome of y:

$$S(y) = yH^\top.$$

Lemma Two words have the same syndrome iff they are in the same coset.

Syndrom decoding Assume that a standard array of an $[n, k]$ code C is given and, in addition, let in the last $n - k$ columns the syndrome for each coset be given.

$$\begin{array}{cccc|cccc|cccc|cccc|cc} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \end{array}$$

When a word y is received, then compute $S(y) = yH^\top$, then locate $S(y)$ in the "syndrome column".

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n, k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left| I_4 \begin{vmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix} \right| \Rightarrow \text{parity check m. } H = \left| \begin{vmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{vmatrix} I_3 \right|$$

Definition Suppose H is a parity-check matrix of an $[n, k]$-code C. Then for any $y \in F_q^n$ the following word is called the syndrome of y:

$$S(y) = yH^\top.$$

Lemma Two words have the same syndrome iff they are in the same coset.

Syndrom decoding Assume that a standard array of an $[n, k]$ code C is given and, in addition, let in the last $n - k$ columns the syndrome for each coset be given.

$$\begin{array}{cccc|cccc|cccc|cccc|cc}
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0
\end{array}$$

When a word y is received, then compute $S(y) = yH^\top$, then locate $S(y)$ in the "syndrome column". Afterwords locate y in the same row

SYNDROME DECODING

Theorem If $G = [I_k|A]$ is the standard form generator matrix of an $[n, k]$-code C, then a parity check matrix for C is $H = [A^\top|I_{n-k}]$.

Example

$$\text{Generator matrix } G = \left| I_4 \begin{vmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix} \right| \Rightarrow \text{parity check m. } H = \left| \begin{vmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{vmatrix} I_3 \right|$$

Definition Suppose H is a parity-check matrix of an $[n, k]$-code C. Then for any $y \in F_q^n$ the following word is called the syndrome of y:

$$S(y) = yH^\top.$$

Lemma Two words have the same syndrome iff they are in the same coset.

Syndrom decoding Assume that a standard array of an $[n, k]$ code C is given and, in addition, let in the last $n - k$ columns the syndrome for each coset be given.

$$\begin{array}{cccc|cccc|cccc|cccc|cc} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \end{array}$$

When a word y is received, then compute $S(y) = yH^\top$, then locate $S(y)$ in the "syndrome column". Afterwords locate y in the same row and decode y as the codeword in the same column and in the first row.

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.
- **Step 2** Locate $z = S(y)$ in the syndrome column.

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.
- **Step 2** Locate $z = S(y)$ in the syndrome column.
- **Step 3** Decode y as $y - l(z)$.

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.
- **Step 2** Locate $z = S(y)$ in the syndrome column.
- **Step 3** Decode y as $y - l(z)$.

Example If $y = 1111$, then $S(y) = 01$ and the above decoding procedure produces

$$1111–0100 = 1011.$$

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.
- **Step 2** Locate $z = S(y)$ in the syndrome column.
- **Step 3** Decode y as $y - l(z)$.

Example If $y = 1111$, then $S(y) = 01$ and the above decoding procedure produces

$$1111\text{–}0100 = 1011.$$

Syndrom decoding is much faster than searching for a nearest codeword to a received word.

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a “syndrome decoding” it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.
- **Step 2** Locate $z = S(y)$ in the syndrome column.
- **Step 3** Decode y as $y - l(z)$.

Example If $y = 1111$, then $S(y) = 01$ and the above decoding procedure produces

$$1111–0100 = 1011.$$

Syndrom decoding is much faster than searching for a nearest codeword to a received word. However, for large codes it is still too inefficient to be practical.

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a “syndrome decoding” it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.
- **Step 2** Locate $z = S(y)$ in the syndrome column.
- **Step 3** Decode y as $y - l(z)$.

Example If $y = 1111$, then $S(y) = 01$ and the above decoding procedure produces

$$1111–0100 = 1011.$$

Syndrom decoding is much faster than searching for a nearest codeword to a received word. However, for large codes it is still too inefficient to be practical.

In general, the problem of finding the nearest neighbour in a linear code is NP-complete.

KEY OBSERVATION for SYNDROM COMPUTATION

When preparing a "syndrome decoding" it is sufcient to store only two columns: one for **coset leaders** and one for **syndromes**.

Example

coset leaders	syndromes
l(z)	z
0000	00
1000	11
0100	01
0010	10

Decoding procedure

- **Step 1** Given y compute $S(y)$.
- **Step 2** Locate $z = S(y)$ in the syndrome column.
- **Step 3** Decode y as $y - l(z)$.

Example If $y = 1111$, then $S(y) = 01$ and the above decoding procedure produces

$$1111–0100 = 1011.$$

Syndrom decoding is much faster than searching for a nearest codeword to a received word. However, for large codes it is still too inefficient to be practical.

In general, the problem of finding the nearest neighbour in a linear code is NP-complete. Fortunately, there are important linear codes with really efficient decoding.

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

Definition Let r be an integer and H be an $r \times (2^r - 1)$ matrix columns of which are all non-zero distinct words from F_2^r. The code having H as its parity-check matrix is called binary Hamming code and denoted by $Ham(r, 2)$.

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

Definition Let r be an integer and H be an $r \times (2^r - 1)$ matrix columns of which are all non-zero distinct words from F_2^r. The code having H as its parity-check matrix is called binary Hamming code and denoted by $Ham(r, 2)$.

Example

$$Ham(2,2) : H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

Definition Let r be an integer and H be an $r \times (2^r - 1)$ matrix columns of which are all non-zero distinct words from F_2^r. The code having H as its parity-check matrix is called binary Hamming code and denoted by $Ham(r, 2)$.

Example

$$Ham(2,2) : H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$Ham(3,2) = H = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

Definition Let r be an integer and H be an $r \times (2^r - 1)$ matrix columns of which are all non-zero distinct words from F_2^r. The code having H as its parity-check matrix is called binary Hamming code and denoted by $Ham(r, 2)$.

Example

$$Ham(2,2) : H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$Ham(3,2) = H = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Theorem Hamming code $Ham(r, 2)$

- is $[2^r - 1, 2^r - 1 - r]$-code,

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

Definition Let r be an integer and H be an $r \times (2^r - 1)$ matrix columns of which are all non-zero distinct words from F_2^r. The code having H as its parity-check matrix is called binary Hamming code and denoted by $Ham(r, 2)$.

Example

$$Ham(2,2) : H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$Ham(3,2) = H = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Theorem Hamming code $Ham(r, 2)$

- is $[2^r - 1, 2^r - 1 - r]$-code,
- has minimum distance 3,

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

Definition Let r be an integer and H be an $r \times (2^r - 1)$ matrix columns of which are all non-zero distinct words from F_2^r. The code having H as its parity-check matrix is called binary Hamming code and denoted by $Ham(r, 2)$.

Example

$$Ham(2,2) : H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$Ham(3,2) = H = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Theorem Hamming code $Ham(r, 2)$

- is $[2^r - 1, 2^r - 1 - r]$-code,
- has minimum distance 3,
- and is a perfect code.

HAMMING CODES

An important family of simple linear codes that are easy to encode and decode, are so-called Hamming codes.

Definition Let r be an integer and H be an $r \times (2^r - 1)$ matrix columns of which are all non-zero distinct words from F_2^r. The code having H as its parity-check matrix is called binary Hamming code and denoted by $Ham(r,2)$.

Example

$$Ham(2,2) : H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$Ham(3,2) = H = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Theorem Hamming code $Ham(r,2)$

- is $[2^r - 1, 2^r{-}1 - r]$-code,
- has minimum distance 3,
- and is a perfect code.

Properties of binary Hamming codes Coset leaders are precisely words of weight ≤ 1. The syndrome of the word $0 \ldots 010 \ldots 0$ with 1 in j-th position and 0 otherwise is the transpose of the j-th column of H.

HAMMING CODES - DECODING

Decoding algorithm for the case the columns of H are arranged in the order of increasing binary numbers the columns represent.

Decoding algorithm for the case the columns of H are arranged in the order of increasing binary numbers the columns represent.

- **Step 1** Given y compute syndrome $S(y) = yH^\top$.

Decoding algorithm for the case the columns of H are arranged in the order of increasing binary numbers the columns represent.

- **Step 1** Given y compute syndrome $S(y) = yH^{\top}$.
- **Step 2** If $S(y) = 0$, then y is assumed to be the codeword sent.

Decoding algorithm for the case the columns of H are arranged in the order of increasing binary numbers the columns represent.

- **Step 1** Given y compute syndrome $S(y) = yH^{\top}$.
- **Step 2** If $S(y) = 0$, then y is assumed to be the codeword sent.
- **Step 3** If $S(y) \neq 0$, then assuming a single error, $S(y)$ gives the binary position of the error.

EXAMPLE

EXAMPLE

For the Hamming code given by the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

EXAMPLE

For the Hamming code given by the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and the received word

$$y = 1101011,$$

EXAMPLE

For the Hamming code given by the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and the received word

$$y = 1101011,$$

we get syndrome

$$S(y) = 110$$

EXAMPLE

For the Hamming code given by the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and the received word

$$y = 1101011,$$

we get syndrome

$$S(y) = 110$$

and therefore the error is in the sixth position.

EXAMPLE

For the Hamming code given by the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and the received word

$$y = 1101011,$$

we get syndrome

$$S(y) = 110$$

and therefore the error is in the sixth position.

Hamming code was discovered by Hamming (1950), Golay (1950).

EXAMPLE

For the Hamming code given by the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and the received word

$$y = 1101011,$$

we get syndrome

$$S(y) = 110$$

and therefore the error is in the sixth position.

Hamming code was discovered by Hamming (1950), Golay (1950).

It was conjectured for some time that Hamming codes and two so called Golay codes are the only non-trivial perfect codes.

Comment

EXAMPLE

For the Hamming code given by the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and the received word

$$y = 1101011,$$

we get syndrome

$$S(y) = 110$$

and therefore the error is in the sixth position.

Hamming code was discovered by Hamming (1950), Golay (1950).

It was conjectured for some time that Hamming codes and two so called Golay codes are the only non-trivial perfect codes.

Comment

Hamming codes were originally used to deal with errors in long-distance telephon calls.

ADVANTAGES of HAMMING CODES

Let a binary symmetric channel be used which with probability q correctly transfers a binary symbol.

ADVANTAGES of HAMMING CODES

Let a binary symmetric channel be used which with probability q correctly transfers a binary symbol.

If a 4-bit message is transmitted through such a channel, then correct transmission of the message occurs with probability q^4.

ADVANTAGES of HAMMING CODES

Let a binary symmetric channel be used which with probability q correctly transfers a binary symbol.

If a 4-bit message is transmitted through such a channel, then correct transmission of the message occurs with probability q^4.

If Hamming $(7, 4, 3)$ code is used to transmit a 4-bit message, then probability of correct decoding is

$$q^7 + 7(1 - q)q^6.$$

ADVANTAGES of HAMMING CODES

Let a binary symmetric channel be used which with probability q correctly transfers a binary symbol.

If a 4-bit message is transmitted through such a channel, then correct transmission of the message occurs with probability q^4.

If Hamming $(7, 4, 3)$ code is used to transmit a 4-bit message, then probability of correct decoding is

$$q^7 + 7(1 - q)q^6.$$

In case $q = 0.9$ the probability of correct transmission is 0.6561 in the case no error correction is used and 0.8503 in the case Hamming code is used - an essential improvement.

SOME BASIC IMPORTANT CODES

SOME BASIC IMPORTANT CODES

- Hamming $(7, 4, 3)$-code. It has 16 codewords of length 7. It can be used to send $2^7 = 128$ messages and can be used to correct 1 error.

SOME BASIC IMPORTANT CODES

- Hamming $(7, 4, 3)$-code. It has 16 codewords of length 7. It can be used to send $2^7 = 128$ messages and can be used to correct 1 error.
- Golay $(23, 12, 7)$-code. It has 4 096 codewords. It can be used to transmit 8 388 608 messages and can correct 3 errors.

SOME BASIC IMPORTANT CODES

- Hamming $(7,4,3)$-code. It has 16 codewords of length 7. It can be used to send $2^7 = 128$ messages and can be used to correct 1 error.
- Golay $(23,12,7)$-code. It has 4 096 codewords. It can be used to transmit 8 388 608 messages and can correct 3 errors.
- Quadratic residue $(47,24,11)$-code. It has

16 777 216 codewords

and can be used to transmit

140 737 488 355 238 messages

and correct 5 errors.

SOME BASIC IMPORTANT CODES

- Hamming $(7, 4, 3)$-code. It has 16 codewords of length 7. It can be used to send $2^7 = 128$ messages and can be used to correct 1 error.
- Golay $(23, 12, 7)$-code. It has 4 096 codewords. It can be used to transmit 8 388 608 messages and can correct 3 errors.
- Quadratic residue $(47, 24, 11)$-code. It has

16 777 216 codewords

and can be used to transmit

140 737 488 355 238 messages

and correct 5 errors.

- Hamming and Golay codes are the only non-trivial perfect codes.

SOME BASIC IMPORTANT CODES

- Hamming $(7, 4, 3)$-code. It has 16 codewords of length 7. It can be used to send $2^7 = 128$ messages and can be used to correct 1 error.
- Golay $(23, 12, 7)$-code. It has 4 096 codewords. It can be used to transmit 8 388 608 messages and can correct 3 errors.
- Quadratic residue $(47, 24, 11)$-code. It has

16 777 216 codewords

and can be used to transmit

140 737 488 355 238 messages

and correct 5 errors.

- Hamming and Golay codes are the only non-trivial perfect codes. They are also special cases of quadratic residue codes.

GOLAY CODES - DESCRIPTION

GOLAY CODES - DESCRIPTION

Golay codes G_{24} and G_{23} were used by Voyager I and Voyager II to transmit color pictures of Jupiter and Saturn.

GOLAY CODES - DESCRIPTION

Golay codes G_{24} and G_{23} were used by Voyager I and Voyager II to transmit color pictures of Jupiter and Saturn. **Generation matrix for G_{24} has the following very simple form:**

GOLAY CODES - DESCRIPTION

Golay codes G_{24} and G_{23} were used by Voyager I and Voyager II to transmit color pictures of Jupiter and Saturn. **Generation matrix for G_{24} has the following very simple form:**

$$G = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}$$

GOLAY CODES - DESCRIPTION

Golay codes G_{24} and G_{23} were used by Voyager I and Voyager II to transmit color pictures of Jupiter and Saturn. **Generation matrix for G_{24} has the following very simple form:**

$$G = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&1&1&1&0&0&0&1&0\\
0&1&0&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&0&0&1\\
0&0&1&0&0&0&0&0&0&0&0&0&1&1&0&1&1&0&1&1&1&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&1&0&1&1&0&1&1&1&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&1&0&1&1&0&1&1&1&0\\
0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&1&0&1&1&0&1&1&1\\
0&0&0&0&0&0&1&0&0&0&0&0&1&1&0&0&0&1&0&1&1&0&1&1\\
0&0&0&0&0&0&0&1&0&0&0&0&1&1&1&0&0&0&1&0&1&1&0&1\\
0&0&0&0&0&0&0&0&1&0&0&0&1&1&1&1&0&0&0&1&0&1&1&0\\
0&0&0&0&0&0&0&0&0&1&0&0&1&0&1&1&1&0&0&0&1&0&1&1\\
0&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&0&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&1&1&1&0&0&0&1&0
\end{pmatrix}$$

G_{24} is $(24, 12, 8)$-code and the weights of all codewords are multiples of 4. G_{23} is obtained from G_{24} by deleting last symbols of each codeword of G_{24}. G_{23} is $(23, 12, 7)$-code.

GOLAY CODES - CONSTRUCTION

Matrix G for Golay code G_{24} has actually a simple and regular construction.

GOLAY CODES - CONSTRUCTION

Matrix G for Golay code G_{24} has actually a simple and regular construction.

The first 12 columns are formed by a unitary matrix I_{12}, next column has all 1's.

Matrix G for Golay code G_{24} has actually a simple and regular construction.

The first 12 columns are formed by a unitary matrix I_{12}, next column has all 1's.

Rows of the last 11 columns are cyclic permutations of the first row which has 1 at those positions that are squares modulo 11, that is

$$0,\ 1,\ 3,\ 4,\ 5,\ 9.$$

TWO SIMPLY DEFINED CODES

TWO SIMPLY DEFINED CODES

- **Maximum length code** is $[2^m - 1, m, 2^{m-1}]$-code with the generator matrix whose columns are all binary representations of numbers from 1 to $2^m - 1$.

TWO SIMPLY DEFINED CODES

- **Maximum length code** is $[2^m - 1, m, 2^{m-1}]$-code with the generator matrix whose columns are all binary representations of numbers from 1 to $2^m - 1$.
- **Hadamard code** HC_{2^n}, $n = 1, 2, ...$ is the code with generator matrices defined recursively as

$$M_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

and

- **Maximum length code** is $[2^m - 1, m, 2^{m-1}]$-code with the generator matrix whose columns are all binary representations of numbers from 1 to $2^m - 1$.
- **Hadamard code** HC_{2^n}, $n = 1, 2, \ldots$ is the code with generator matrices defined recursively as

$$M_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

and

$$M_{2^n} = \begin{bmatrix} M_{2^{n-1}} & M_{2^{n-1}} \\ M_{2n-1} & \bar{M}_{2^{n-1}} n \end{bmatrix}$$

where $\bar{M}_n$ is the complementary matrix to M_n (with 0 and 1 interchanged).

EXAMPLE

EXAMPLE

Hadamard code

$$M_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

REED-MULLER CODES

REED-MULLER CODES

This is an infinite, recursively defined, family of so called $RM_{r,m}$ binary linear $[2^m, k, 2^{m-r}]$-codes with

$$k = 1 + \binom{m}{1} + \ldots + \binom{m}{r}.$$

REED-MULLER CODES

This is an infinite, recursively defined, family of so called $RM_{r,m}$ binary linear $[2^m, k, 2^{m-r}]$-codes with

$$k = 1 + \binom{m}{1} + \ldots + \binom{m}{r}.$$

The generator matrix $G_{r,m}$ for $RM_{r,m}$ code has the form

$$G_{r,m} = \begin{bmatrix} G_{r-1,m} \\ Q_r \end{bmatrix}$$

where Q_r is a matrix with dimension $\binom{m}{r} \times 2^m$ where

REED-MULLER CODES

This is an infinite, recursively defined, family of so called $RM_{r,m}$ binary linear $[2^m, k, 2^{m-r}]$-codes with

$$k = 1 + \binom{m}{1} + \ldots + \binom{m}{r}.$$

The generator matrix $G_{r,m}$ for $RM_{r,m}$ code has the form

$$G_{r,m} = \begin{bmatrix} G_{r-1,m} \\ Q_r \end{bmatrix}$$

where Q_r is a matrix with dimension $\binom{m}{r} \times 2^m$ where

- $G_{0,m}$ is a row vector of the length 2^m with all elements 1.

REED-MULLER CODES

This is an infinite, recursively defined, family of so called $RM_{r,m}$ binary linear $[2^m, k, 2^{m-r}]$-codes with

$$k = 1 + \binom{m}{1} + \ldots + \binom{m}{r}.$$

The generator matrix $G_{r,m}$ for $RM_{r,m}$ code has the form

$$G_{r,m} = \begin{bmatrix} G_{r-1,m} \\ Q_r \end{bmatrix}$$

where Q_r is a matrix with dimension $\binom{m}{r} \times 2^m$ where

- $G_{0,m}$ is a row vector of the length 2^m with all elements 1.
- $G_{1,m}$ is obtained from $G_{0,m}$ by adding columns that are binary representations of the column numbers.

REED-MULLER CODES

This is an infinite, recursively defined, family of so called $RM_{r,m}$ binary linear $[2^m, k, 2^{m-r}]$-codes with

$$k = 1 + \binom{m}{1} + \ldots + \binom{m}{r}.$$

The generator matrix $G_{r,m}$ for $RM_{r,m}$ code has the form

$$G_{r,m} = \begin{bmatrix} G_{r-1,m} \\ Q_r \end{bmatrix}$$

where Q_r is a matrix with dimension $\binom{m}{r} \times 2^m$ where

- $G_{0,m}$ is a row vector of the length 2^m with all elements 1.
- $G_{1,m}$ is obtained from $G_{0,m}$ by adding columns that are binary representations of the column numbers.
- Matrix Q_r is obtained by considering all combinations of r rows of $G_{1,m}$ and by obtaining products of these rows/vectors, component by component. The result of each of such a multiplication constitues a row of Q_r.

EXAMPLE

$$G_{1,4} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

and

$$Q_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

EXAMPLE

$$G_{1,4} = \begin{bmatrix} 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1 \\ 0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1 \\ 0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1 \\ 0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1 \end{bmatrix}$$

and

$$Q_2 = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1 \\ 0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1 \\ 0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&1 \\ 0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1 \\ 0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1 \\ 0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1 \end{bmatrix}$$

Codes $R(m-r-1,m)$ and $R(r,m)$ are dual codes.

REED-MULLER CODES II

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \,|\, u + v, \textit{where } u \in D_1, v \in D_2\}$.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \,|\, u + v, where\ u \in D_1, v \in D_2\}$.

Lemma C is $[2n, k_1 + k_2, min\{2d_1, d_2\}]$-code and if G_i is a generator matrix for D_i, $i = 1, 2$, then $\begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}$ is a generator matrix for C.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \,|\, u + v, where\ u \in D_1, v \in D_2\}$.

Lemma C is $[2n, k_1 + k_2, min\{2d_1, d_2\}]$-code and if G_i is a generator matrix for D_i, $i = 1, 2$, then $\begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}$ is a generator matrix for C.

Reed-Muller codes $R(r, m)$, with $0 \leq r \leq m$ are binary codes of length $n = 2^m$.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \mid u + v, where\ u \in D_1, v \in D_2\}$.

Lemma C is $[2n, k_1 + k_2, min\{2d_1, d_2\}]$-code and if G_i is a generator matrix for D_i, $i = 1, 2$, then $\begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}$ is a generator matrix for C.

Reed-Muller codes $R(r, m)$, with $0 \leq r \leq m$ are binary codes of length $n = 2^m$.$R(m, m)$ is the whole set of words of length n, $R(0, m)$ is the repetition code.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \mid u + v, where\ u \in D_1, v \in D_2\}$.

Lemma C is $[2n, k_1 + k_2, min\{2d_1, d_2\}]$-code and if G_i is a generator matrix for D_i, $i = 1, 2$, then $\begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}$ is a generator matrix for C.

Reed-Muller codes $R(r, m)$, with $0 \leq r \leq m$ are binary codes of length $n = 2^m$. $R(m, m)$ is the whole set of words of length n, $R(0, m)$ is the repetition code.

If $0 < r < m$, then $R(r + 1, m + 1)$ is obtained from codes $R(r + 1, m)$ and $R(r, m)$ by the above construction.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \,|\, u + v, \mathit{where}\ u \in D_1, v \in D_2\}$.

Lemma C is $[2n, k_1 + k_2, min\{2d_1, d_2\}]$-code and if G_i is a generator matrix for D_i, $i = 1, 2$, then $\begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}$ is a generator matrix for C.

Reed-Muller codes $R(r, m)$, with $0 \leq r \leq m$ are binary codes of length $n = 2^m$.$R(m, m)$ is the whole set of words of length n, $R(0, m)$ is the repetition code.

If $0 < r < m$, then $R(r + 1, m + 1)$ is obtained from codes $R(r + 1, m)$ and $R(r, m)$ by the above construction.

Theorem The dimension of $R(r, m)$ equals $1 + \binom{m}{1} + \ldots + \binom{m}{r}$.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \,|\, u + v, \textit{where } u \in D_1, v \in D_2\}$.

Lemma C is $[2n, k_1 + k_2, min\{2d_1, d_2\}]$-code and if G_i is a generator matrix for D_i, $i = 1, 2$, then $\begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}$ is a generator matrix for C.

Reed-Muller codes $R(r, m)$, with $0 \leq r \leq m$ are binary codes of length $n = 2^m$.$R(m, m)$ is the whole set of words of length n, $R(0, m)$ is the repetition code.

If $0 < r < m$, then $R(r + 1, m + 1)$ is obtained from codes $R(r + 1, m)$ and $R(r, m)$ by the above construction.

Theorem The dimension of $R(r, m)$ equals $1 + \binom{m}{1} + \ldots + \binom{m}{r}$. The minimum weight of $R(r, m)$ equals 2^{m-r}.

REED-MULLER CODES II

Reed-Muller codes form a family of codes defined recursively with interesting properties and easy decoding.

If D_1 is a binary $[n, k_1, d_1]$-code and D_2 is a binary $[n, k_2, d_2]$-code, a binary code C of length $2n$ is defined as follows $C = \{u \mid u + v, \textit{where } u \in D_1, v \in D_2\}$.

Lemma C is $[2n, k_1 + k_2, min\{2d_1, d_2\}]$-code and if G_i is a generator matrix for D_i, $i = 1, 2$, then $\begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}$ is a generator matrix for C.

Reed-Muller codes $R(r, m)$, with $0 \leq r \leq m$ are binary codes of length $n = 2^m. R(m, m)$ is the whole set of words of length n, $R(0, m)$ is the repetition code.

If $0 < r < m$, then $R(r + 1, m + 1)$ is obtained from codes $R(r + 1, m)$ and $R(r, m)$ by the above construction.

Theorem The dimension of $R(r, m)$ equals $1 + \binom{m}{1} + \ldots + \binom{m}{r}$. The minimum weight of $R(r, m)$ equals 2^{m-r}. Codes $R(m - r - 1, m)$ and $R(r, m)$ are dual codes.

SINGLETON and PLOTKIN BOUNDS

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

Singleton bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}$$

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

Singleton bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}$$

Proof Take some $d-1$ coordinates and project all codewords to the remaining coordinates.

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

Singleton bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}$$

Proof Take some $d-1$ coordinates and project all codewords to the remaining coordinates.

The resulting codewords have to be all different and therefore M cannot be larger than the number of q-ary words of the length $n-d-1$.

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

Singleton bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}$$

Proof Take some $d - 1$ coordinates and project all codewords to the remaining coordinates.

The resulting codewords have to be all different and therefore M cannot be larger than the number of q-ary words of the length $n - d - 1$.

Codes for which $M = q^{n-d+1}$ are called **MDS-codes** (**Maximum Distance Separable**).

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

Singleton bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}$$

Proof Take some $d-1$ coordinates and project all codewords to the remaining coordinates.

The resulting codewords have to be all different and therefore M cannot be larger than the number of q-ary words of the length $n-d-1$.

Codes for which $M = q^{n-d+1}$ are called **MDS-codes** (**Maximum Distance Separable**).

Corollary: If C is a binary linear $[n, k, d]$-code, then

$$d \leq n-k+1.$$

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

Singleton bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}$$

Proof Take some $d-1$ coordinates and project all codewords to the remaining coordinates.

The resulting codewords have to be all different and therefore M cannot be larger than the number of q-ary words of the length $n-d-1$.

Codes for which $M = q^{n-d+1}$ are called **MDS-codes** (**Maximum Distance Separable**).

Corollary: If C is a binary linear $[n, k, d]$-code, then

$$d \leq n - k + 1.$$

So called **Plotkin bound** says

$$d \leq \frac{n2^{k-1}}{2^k - 1}.$$

SINGLETON and PLOTKIN BOUNDS

To determine distance of a linear code can be computationally hard task. For that reason various bounds on distance can be much useful.

Singleton bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}$$

Proof Take some $d-1$ coordinates and project all codewords to the remaining coordinates.

The resulting codewords have to be all different and therefore M cannot be larger than the number of q-ary words of the length $n-d-1$.

Codes for which $M = q^{n-d+1}$ are called **MDS-codes** (**Maximum Distance Separable**).

Corollary: If C is a binary linear $[n, k, d]$-code, then

$$d \leq n - k + 1.$$

So called **Plotkin bound** says

$$d \leq \frac{n2^{k-1}}{2^k - 1}.$$

Plotkin bound implies that q-nary error-correcting codes with $d \geq n(1 - 1/q)$ have only polynomially many codewords and hence are not very interesting.

SHORTENING and PUNCTURING of LINEAR CODES

If C is a q-ary linear $[n, k, d]$-code, then

$D = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, 0) \in C\}$. is a linear code - a shortening of the code C.

SHORTENING and PUNCTURING of LINEAR CODES

If C is a q-ary linear $[n, k, d]$-code, then

$D = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, 0) \in C\}$. is a linear code - a shortening of the code C.

If $d > 1$, then D is a linear $[n-1, k', d^*]$-code, where $k' \in \{k-1, k\}$ and $d^* \geq d$, a so calle shortening of the code C.

SHORTENING and PUNCTURING of LINEAR CODES

If C is a q-ary linear $[n, k, d]$-code, then

$D = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, 0) \in C\}$. is a linear code - a shortening of the code C.

If $d > 1$, then D is a linear $[n-1, k', d^*]$-code, where $k' \in \{k-1, k\}$ and $d^* \geq d$, a so calle shortening of the code C.

If C is a q-ary linear $[n, k, d]$-code and

$$E = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, x) \in C, \textit{for some } x \leq q\},$$

SHORTENING and PUNCTURING of LINEAR CODES

If C is a q-ary linear $[n, k, d]$-code, then

$D = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, 0) \in C\}.$ is a linear code - a shortening of the code C.

If $d > 1$, then D is a linear $[n-1, k', d^*]$-code, where $k' \in \{k-1, k\}$ and $d^* \geq d$, a so calle shortening of the code C.

If C is a q-ary linear $[n, k, d]$-code and

$$E = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, x) \in C, \textit{for some } x \leq q\},$$

then E is a linear code - **a puncturing of the code** C.

SHORTENING and PUNCTURING of LINEAR CODES

If C is a q-ary linear $[n, k, d]$-code, then

$D = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, 0) \in C\}$. is a linear code - a shortening of the code C.

If $d > 1$, then D is a linear $[n-1, k', d^*]$-code, where $k' \in \{k-1, k\}$ and $d^* \geq d$, a so calle shortening of the code C.

If C is a q-ary linear $[n, k, d]$-code and

$$E = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, x) \in C, \textit{for some } x \leq q\},$$

then E is a linear code - **a puncturing of the code** C.

If $d > 1$, then E is an $[n-1, k, d^*]$ code where $d^* = d - 1$ if C has a minimum weight codeword with wit non-zero last coordinate and $d^* = d$ otherwise.

SHORTENING and PUNCTURING of LINEAR CODES

If C is a q-ary linear $[n, k, d]$-code, then

$D = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, 0) \in C\}.$ is a linear code - a shortening of the code C.

If $d > 1$, then D is a linear $[n-1, k', d^*]$-code, where $k' \in \{k-1, k\}$ and $d^* \geq d$, a so calle shortening of the code C.

If C is a q-ary linear $[n, k, d]$-code and

$$E = \{(x_1, \ldots, x_{n-1}) | (x_1, \ldots, x_{n-1}, x) \in C, \textit{for some } x \leq q\},$$

then E is a linear code - **a puncturing of the code** C.

If $d > 1$, then E is an $[n-1, k, d^*]$ code where $d^* = d - 1$ if C has a minimum weight codeword with wit non-zero last coordinate and $d^* = d$ otherwise.

When $d = 1$, then E is an $[n-1, k, 1]$ code, if C has no codeword of weight 1 whose nonzero entry is in last coordinate; otherwise, if $k > 1$, then E is an $[n-1, k-1, d^*]$ code with $d^* > 1$

REED-SOLOMON CODES

REED-SOLOMON CODES

An important example of MDS-codes are q-ary Reed-Solomon codes RSC(k, q), for $k \leq q$.

They are codes a generator matrix of which has rows labelled by polynomials X^i, $0 \leq i \leq k-1$, columns labeled by elements $0, 1, \ldots, q-1$ and the element in the row labelled by a polynomial p and in the column labelled by an element u is $p(u)$.

REED-SOLOMON CODES

An important example of MDS-codes are q-ary Reed-Solomon codes RSC(k, q), for $k \leq q$.

They are codes a generator matrix of which has rows labelled by polynomials X^i, $0 \leq i \leq k-1$, columns labeled by elements $0, 1, \ldots, q-1$ and the element in the row labelled by a polynomial p and in the column labelled by an element u is $p(u)$.

RSC(k, q) code is $[q, k, q-k+1]$ code.

REED-SOLOMON CODES

An important example of MDS-codes are q-ary Reed-Solomon codes RSC(k, q), for $k \leq q$.

They are codes a generator matrix of which has rows labelled by polynomials X^i, $0 \leq i \leq k-1$, columns labeled by elements $0, 1, \ldots, q-1$ and the element in the row labelled by a polynomial p and in the column labelled by an element u is $p(u)$.

RSC(k, q) code is $[q, k, q-k+1]$ code.

Example Generator matrix for RSC$(3, 5)$ code is

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 4 & 4 & 1 \end{bmatrix}$$

REED-SOLOMON CODES

An important example of MDS-codes are q-ary Reed-Solomon codes RSC(k, q), for $k \leq q$.

They are codes a generator matrix of which has rows labelled by polynomials X^i, $0 \leq i \leq k-1$, columns labeled by elements $0, 1, \ldots, q-1$ and the element in the row labelled by a polynomial p and in the column labelled by an element u is $p(u)$.

RSC(k, q) code is $[q, k, q-k+1]$ code.

Example Generator matrix for RSC$(3, 5)$ code is

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 4 & 4 & 1 \end{bmatrix}$$

Interesting property of Reed-Solomon codes:

$$\text{RSC}(k, q)^{\perp} = \text{RSC}(q-k, q).$$

REED-SOLOMON CODES

An important example of MDS-codes are q-ary Reed-Solomon codes RSC(k, q), for $k \leq q$.

They are codes a generator matrix of which has rows labelled by polynomials X^i, $0 \leq i \leq k-1$, columns labeled by elements $0, 1, \ldots, q-1$ and the element in the row labelled by a polynomial p and in the column labelled by an element u is $p(u)$.

RSC(k, q) code is $[q, k, q-k+1]$ code.

Example Generator matrix for RSC$(3, 5)$ code is

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 1 & 4 & 4 & 1 \end{bmatrix}$$

Interesting property of Reed-Solomon codes:

$$\text{RSC}(k, q)^{\perp} = \text{RSC}(q-k, q).$$

Reed-Solomon codes are used in digital television, satellite communication, wireless communication, barcodes, compact discs, DVD,... They are very good to correct burst errors - such as ones caused by solar energy.

Ternary Golay code with parameters (11, 729, 5) can be used to bet for results of 11 soccer games with potential outcomes 1 (if home team wins), 2 (if guest team wins) and 3 (in case of a draw).

SOCCER GAMES BETTING SYSTEM

Ternary Golay code with parameters (11, 729, 5) can be used to bet for results of 11 soccer games with potential outcomes 1 (if home team wins), 2 (if guest team wins) and 3 (in case of a draw).

If 729 bets are made, then at least one bet has at least 9 results correctly guessed.

SOCCER GAMES BETTING SYSTEM

Ternary Golay code with parameters (11, 729, 5) can be used to bet for results of 11 soccer games with potential outcomes 1 (if home team wins), 2 (if guest team wins) and 3 (in case of a draw).

If 729 bets are made, then at least one bet has at least 9 results correctly guessed.

In case one has to bet for 13 games, then one can usually have two games with pretty sure outcomes and for the rest one can use the above ternary Golay code.

APPENDIX

LDPC (Low-Density Parity Check) - CODES

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

A linear $[n, k]$ code is said to be a **regular** $[n, k, r, c]$ **LDPC code** if $r << n, c << n - k$ and its parity-check matrix has exactly r 1's in each row and exactly c 1's in each column.

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

A linear $[n, k]$ code is said to be a **regular** $[n, k, r, c]$ **LDPC code** if $r << n, c << n - k$ and its parity-check matrix has exactly r 1's in each row and exactly c 1's in each column.

In the recent years LDPC codes are replacing in many important applications other types of codes for the following reasons:

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

A linear $[n, k]$ code is said to be a **regular** $[n, k, r, c]$ **LDPC code** if $r << n, c << n - k$ and its parity-check matrix has exactly r 1's in each row and exactly c 1's in each column.

In the recent years LDPC codes are replacing in many important applications other types of codes for the following reasons:

1. LDPC codes are in principle also very good channel codes, so called **Shannon capacity approaching codes**, they allow the noise threshold to be set arbitrarily close to the theoretical maximum - to Shannon limit - for symmetric channel.

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

A linear $[n, k]$ code is said to be a **regular** $[n, k, r, c]$ **LDPC code** if $r << n, c << n - k$ and its parity-check matrix has exactly r 1's in each row and exactly c 1's in each column.

In the recent years LDPC codes are replacing in many important applications other types of codes for the following reasons:

1. LDPC codes are in principle also very good channel codes, so called **Shannon capacity approaching codes**, they allow the noise threshold to be set arbitrarily close to the theoretical maximum - to Shannon limit - for symmetric channel.
2. Good LDPC codes can be decoded in time linear to their block length using special (for example "iterative belief propagation") approximation techniques.

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

A linear $[n, k]$ code is said to be a **regular** $[n, k, r, c]$ **LDPC code** if $r << n, c << n - k$ and its parity-check matrix has exactly r 1's in each row and exactly c 1's in each column.

In the recent years LDPC codes are replacing in many important applications other types of codes for the following reasons:

1. LDPC codes are in principle also very good channel codes, so called **Shannon capacity approaching codes**, they allow the noise threshold to be set arbitrarily close to the theoretical maximum - to Shannon limit - for symmetric channel.
2. Good LDPC codes can be decoded in time linear to their block length using special (for example "iterative belief propagation") approximation techniques.
3. Some LDPC codes are well suited for implementations that make heavy use of parallelism.

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

A linear $[n, k]$ code is said to be a **regular** $[n, k, r, c]$ **LDPC code** if $r << n, c << n - k$ and its parity-check matrix has exactly r 1's in each row and exactly c 1's in each column.

In the recent years LDPC codes are replacing in many important applications other types of codes for the following reasons:

1. LDPC codes are in principle also very good channel codes, so called **Shannon capacity approaching codes**, they allow the noise threshold to be set arbitrarily close to the theoretical maximum - to Shannon limit - for symmetric channel.
2. Good LDPC codes can be decoded in time linear to their block length using special (for example "iterative belief propagation") approximation techniques.
3. Some LDPC codes are well suited for implementations that make heavy use of parallelism.

LDPC (Low-Density Parity Check) - CODES

A LDPC code is a binary linear code whose parity check matrix is very sparse - it contains only very few 1's.

A linear $[n, k]$ code is said to be a **regular** $[n, k, r, c]$ **LDPC code** if $r << n, c << n - k$ and its parity-check matrix has exactly r 1's in each row and exactly c 1's in each column.

In the recent years LDPC codes are replacing in many important applications other types of codes for the following reasons:

1. LDPC codes are in principle also very good channel codes, so called **Shannon capacity approaching codes**, they allow the noise threshold to be set arbitrarily close to the theoretical maximum - to Shannon limit - for symmetric channel.
2. Good LDPC codes can be decoded in time linear to their block length using special (for example "iterative belief propagation") approximation techniques.
3. Some LDPC codes are well suited for implementations that make heavy use of parallelism.

Parity-check matrices for LDPC codes are often (pseudo)-randomly generated, subject to sparsity constrains. Such LDPC codes are proven to be good with a high probability.

DISCOVERY and APPLICATION of LDPC CODES

LDPC codes were discovered in 1960 by R.C. Gallager in his PhD thesis,

DISCOVERY and APPLICATION of LDPC CODES

LDPC codes were discovered in 1960 by R.C. Gallager in his PhD thesis, but were ignored till 1996 when linear time decoding methods were discovered for some of them.

DISCOVERY and APPLICATION of LDPC CODES

LDPC codes were discovered in 1960 by R.C. Gallager in his PhD thesis, but were ignored till 1996 when linear time decoding methods were discovered for some of them.

LDPC codes are used for: deep space communication; digital video broadcasting; 10GBase-T Ethernet, which sends data at 10 gigabits per second over Twisted-pair cables; Wi-Fi standard,....

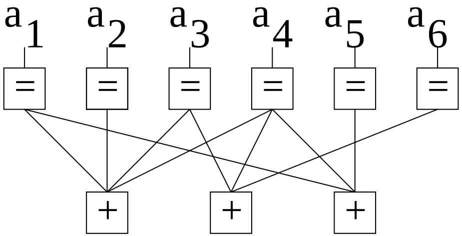
DISCOVERY and APPLICATION of LDPC CODES

LDPC codes were discovered in 1960 by R.C. Gallager in his PhD thesis, but were ignored till 1996 when linear time decoding methods were discovered for some of them.

LDPC codes are used for: deep space communication; digital video broadcasting; 10GBase-T Ethernet, which sends data at 10 gigabits per second over Twisted-pair cables; Wi-Fi standard,....

BI-PARTITE (TANNER) GRAPHS REPRESENTATION of LDPC CODES

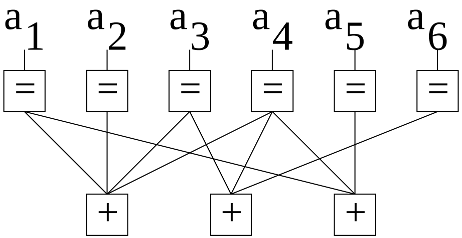
An $[n, k]$ LDPC code can be represented by a bipartite graph between a set of n top "variable-nodes (v-nodes)" and a set of bottom $(n - k)$ "parity check nodes (pc-nodes)".
Variable nodes:



Parity check nodes:

BI-PARTITE (TANNER) GRAPHS REPRESENTATION of LDPC CODES

An $[n, k]$ LDPC code can be represented by a bipartite graph between a set of n top "variable-nodes (v-nodes)" and a set of bottom $(n-k)$ "parity check nodes (pc-nodes)".
Variable nodes:



Parity check nodes:

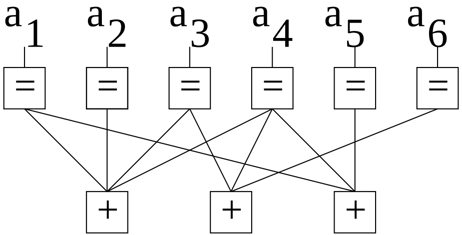
The corresponding parity check matrix has $n-k$ rows and n columns and i-th column has 1 in the j-th row exactly in case if i-th v-node is connected to j-th c-node.

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

TANNER GRAPHS - CONTINUATION

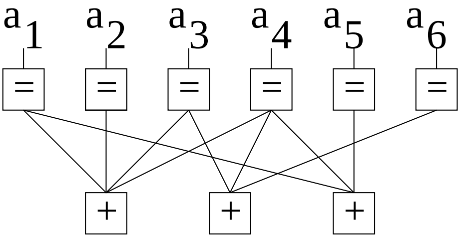
TANNER GRAPHS - CONTINUATION

The LDPC-code with the Tanner bipartite graph for $(6, 3)$ LDPC-code.



TANNER GRAPHS - CONTINUATION

The LDPC-code with the Tanner bipartite graph for $(6,3)$ LDPC-code.

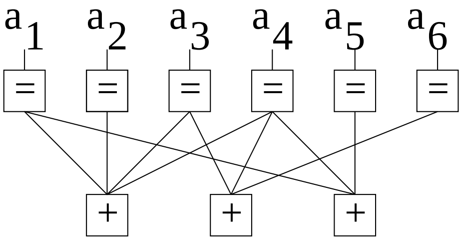


has the parity check matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

TANNER GRAPHS - CONTINUATION

The LDPC-code with the Tanner bipartite graph for $(6, 3)$ LDPC-code.



has the parity check matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

and therefore the following constrains have to be satisfied:

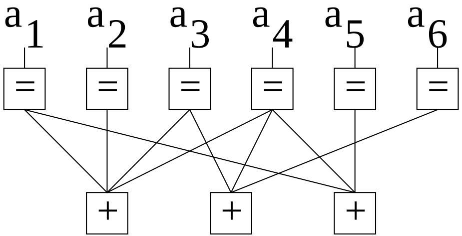
$$a_1 + a_2 + a_3 + a_4 = 0$$

$$a_3 + a_4 + a_6 = 0$$

$$a_1 + a_4 + a_5 = 0$$

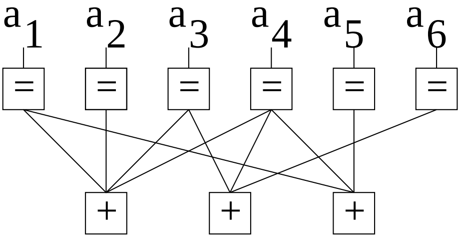
DECODING

Since for the LDPC-code with the Tanner bipartite graph for $(6, 3)$ LDPC-code.



DECODING

Since for the LDPC-code with the Tanner bipartite graph for $(6,3)$ LDPC-code.



the following constrains have to be satisfied:

$$a_1 + a_2 + a_3 + a_4 = 0$$

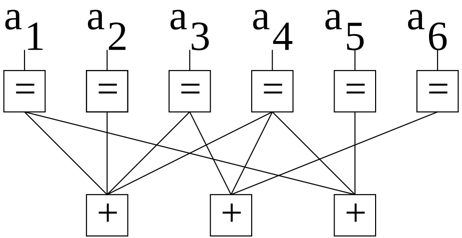
$$a_3 + a_4 + a_6 = 0$$

$$a_1 + a_4 + a_5 = 0$$

Let the word ?01?11 be received.

DECODING

Since for the LDPC-code with the Tanner bipartite graph for $(6,3)$ LDPC-code.



the following constrains have to be satisfied:

$$a_1 + a_2 + a_3 + a_4 = 0$$

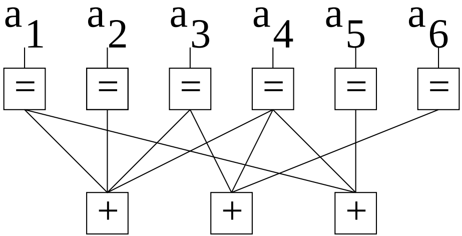
$$a_3 + a_4 + a_6 = 0$$

$$a_1 + a_4 + a_5 = 0$$

Let the word ?01?11 be received.From the second equation it follows that the second unknown symbol is

DECODING

Since for the LDPC-code with the Tanner bipartite graph for $(6,3)$ LDPC-code.



the following constrains have to be satisfied:

$$a_1 + a_2 + a_3 + a_4 = 0$$

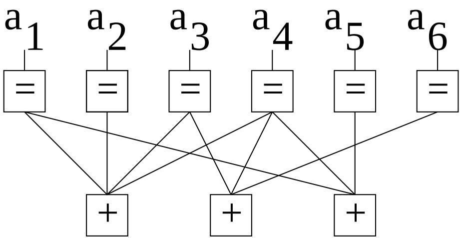
$$a_3 + a_4 + a_6 = 0$$

$$a_1 + a_4 + a_5 = 0$$

Let the word ?01?11 be received.From the second equation it follows that the second unknown symbol is 0. From the last equation it then follows that the first unknown symbol is

DECODING

Since for the LDPC-code with the Tanner bipartite graph for $(6,3)$ LDPC-code.



the following constrains have to be satisfied:

$$a_1 + a_2 + a_3 + a_4 = 0$$

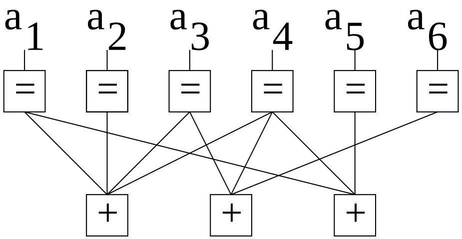
$$a_3 + a_4 + a_6 = 0$$

$$a_1 + a_4 + a_5 = 0$$

Let the word ?01?11 be received.From the second equation it follows that the second unknown symbol is 0. From the last equation it then follows that the first unknown symbol is 1.

DECODING

Since for the LDPC-code with the Tanner bipartite graph for $(6,3)$ LDPC-code.



the following constrains have to be satisfied:

$$a_1 + a_2 + a_3 + a_4 = 0$$

$$a_3 + a_4 + a_6 = 0$$

$$a_1 + a_4 + a_5 = 0$$

Let the word ?01?11 be received.From the second equation it follows that the second unknown symbol is 0. From the last equation it then follows that the first unknown symbol is 1.

Using so called **iterative belief propagation techniques**, LDPC codes can be decoded in time linear to their block length.

DESIGN of LDPC codes

DESIGN of LDPC codes

- Some good LDPC codes were designed through randomly chosen parity check matrices.
- Some LDPC codes are based on Reed-Solomon codes, such as the RS-LDPC code used in the 10-gigabit Ethernet standard.