Applications of Finite Sets

Jeremy Knight
Final Oral Exam
Texas A&M University
March 29th 2012

Finite Fields and Cryptography

A field is a set that

1. is associative, commutative, and distributive for *addition* and *multiplication*,
2. contains an additive identity element (zero) and multiplicative identity element (unity),
3. contains an additive inverse for all elements, and
4. contains a multiplicative inverse for all non-zero elements.

Basics of Finite Fields

- A *finite field* is field that has a finite number of elements.
- **Order:** the number of elements in a field
- The order *must be of the form p^n* for some prime number p and integer $n > 1$.
- Standard Notation: $GF(p^n)$ where the "GF" represents "Galois Field" in honor of Evariste Galois.
- In cryptographic systems, it is common to apply the field $GF(2^n)$ and work modulo 2 to work with modern computers.

Constructing $GF(2^m)$

- $\boldsymbol{Z}_p[X]$: the set of polynomials with coefficients mod p.
- We will typically work with polynomials in $\boldsymbol{Z}_2[X]$ which we often represent it in binary notation.
- For example,
  $\mathrm{X}^8 + \mathrm{X}^4 + \mathrm{X}^3 + \mathrm{X} + 1 \rightarrow 100010011$
  (an important polynomial for the Advanced Encryption Standard (AES).)
- The binary digits $b_8 b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$ are the coefficients of $b_8 X^8 + \cdots + b_1 X^1 + b_0$.

Arithmetic of $GF(2^m)$

Addition and Subtraction

- Addition is the XOR operation, denoted with the symbol $\oplus$ modulo 2.
- $1 \oplus 1 = 0,\ 1 \oplus 0 = 1,\ 0 \oplus 0 = 0$
- *Example* : Add
  $(X^8 + X^4 + X^3 + X + 1) + (X^8 + X^7 + X^3 + 1)$
  as a polynomial and in binary notation.
  $(X^8 + X^4 + X^3 + X + 1) + (X^8 + X^7 + X^3 + 1)$
  $= X^7 + X^4 + X$

Note: the $X^8, X^3,$ and1 terms have vanished since the coefficients are $2 \equiv 0$ (mod2).

Arithmetic of $GF(2^m)$

- *Example (cont.)*

In binary notation this sum is
$\{100011011\} \oplus \{110001001\} = \{010010010\}$.

- Note: subtraction of polynomials in $Z_2[X]$ is equivalent to addition since $-1 \equiv 1 \pmod 2$
  $$a - b \equiv a + (-1)b \equiv a + b \pmod 2$$
  for all $a, b \equiv 0$ *or* $1 \pmod 2$.

Arithmetic of $GF(2^m)$

Multiplication

- Multiplication of polynomials in $Z_2[X]$ is done in the normal manner applying distribution.
- Some powers of X will vanish in mod 2.

- *Example* Compute $(X^2+X+1)(X+1)$ as a polynomial and in binary notation.

$$(X^2+X+1)(X+1)$$
$$=(X^3+X^2+X)+(X^2+X+1)$$
$$=X^3+1$$

Arithmetic of $GF(2^m)$

Multiplication

Example (cont.)

In binary notation:

$$\{0111\}\cdot\{0011\}$$
$$=\{0111\}\cdot\{0010\}\oplus\{0111\}\cdot\{0001\}$$
$$=\{1110\}\oplus\{0111\}=\{1001\}$$

- Note: multiplying $\{0111\}$ by $\{0010\}$ shifts all the bits to the left on place and adds a 0 to the right.
- Hence, binary multiplication is simply a series of "bit shifts" and XOR operations.

Arithmetic of $GF(2^m)$

Multiplication

Example Compute the product of the 8-bit binary numbers $\{10011011\}\cdot\{00100101\}$

$$\{10011011\}\cdot\{00100101\}$$
$$=\{10011011\}\cdot\{00100000\}$$
$$\oplus\{10011011\}\cdot\{00000100\}$$
$$\oplus\{10011011\}\cdot\{00000001\}$$
$$=\{1001101100000\}$$
$$\oplus\{0001001101100\}$$
$$\oplus\{0000010011011\}$$
$$=\{1000110010111\}$$

Arithmetic of $GF(2^m)$

Division

- Method 1: long division in $Z_2[X]$.
- *Example* Use long division to divide X^4+1 by X^2+X+1

$$\begin{array}{r|l} & X^2+X \quad r.\ X+1 \\ \hline X^2+X+1 & X^4+1 \\ & \underline{X^4+X^3+X^2} \\ & \quad X^3+X^2+1 \\ & \quad \underline{X^3+X^2+X} \\ & \qquad\qquad X+1 \end{array}$$

Arithmetic of $GF(2^m)$

Division

$$X^4+1=(X^2+X)(X^2+X+1)+(X+1)$$
$$or$$
$$X^4+1\equiv X+1(\text{mod } X^2+X+1)$$

- Method 2: Binary Division
- *Example* Use binary notation to divide X^4+1 by X^2+X+1

Arithmetic of $GF(2^m)$

Division

- *Example* Use binary notation to divide X^4+1 by X^2+X+1

$$\frac{10001}{111}=\frac{11100\oplus 1101}{111}$$
$$=100\oplus\frac{1101}{111}$$
$$=100\oplus\frac{1110\oplus 11}{111}$$
$$=100\oplus 10\oplus\frac{11}{111}$$
$$=110\oplus\frac{11}{111}$$
$$\text{or } X^2+X+\frac{X+1}{X^2+X+1}$$

Arithmetic of $GF(2^m)$

MATLAB Algorithms.

- *binxor.m :*
  Binary addition is performed in one line in c=dec2bin(bitxor(bin2dec(a),bin2dec(b)));
- *binmult.m:*
  Binary multiplication applying a bitshift and distribution.
- *bindiv.m:*
  Binary division
- *bin2poly.m:*
  Converts a binary number to a polynomial.

Irreducible Polynomials

- For small values of n we can check all products of polynomials in $Z_{n-1}[X]$ to find a polynomial that is irreducible.
- Consider the nonzero elements of $GF(2^3)$

$$X^2,\quad X^2+X,\quad X^2+1,\quad X^2+X+1,$$
$$X,\quad X+1,\quad 1$$

Irreducible Polynomials

- Irreducible polynomial:
  $P(X) \in \boldsymbol{Z}_p[X]$ that does not factor into polynomials of lower degree mod 2.
- Used to construct a finite field with p^n elements for prime p and integer $n \geq 1$ by working modulo $P(X)$ for irreducible $P(X)$.
- Consider the possible 2nd degree polynomials in $\boldsymbol{Z}_2[X]$.
  $X^2,\quad X^2+1,\quad X^2+X,\quad X^2+X+1$
- Three of these can be factored into polynomials in $\boldsymbol{Z}_2[X]$ as
  $$X \cdot X = X^2$$
  $$X \cdot (X+1) = X^2+X$$
  $$(X+1)\cdot(X+1) = X^2+1$$
- X^2+X+1 is irreducible.

Irreducible Polynomials

- We will check all products that produce a polynomial of degree 3.

$X^2(X) = X^3$
$X^2(X+1) = X^3+X$
$(X^2+X)(X) = X^3+X^2$
$(X^2+X)(X+1) = X^3+X^2+X^2+X = X^3+X$
$(X^2+1)(X) = X^3+X$
$(X^2+1)(X+1) = X^3+X^2+X+1$
$(X^2+X+1)(X) = X^3+X^2+X$
$(X^2+X+1)(X+1) = X^3+X^2+X+X^2+X+1$
$= X^3+1$

Irreducible Polynomials

- We observe that the only $Z_2[X]$ polynomials of degree 3 that are not produced above are
  $f(X) = X^3+X^2+1$, and
  $f(X) = X^3+X+1$.
- Thus, these are irreducible polynomials in $GF(2^3)$

Multiplicative Inverse

- When working with $GF(2^m)$ modulo an irreducible polynomial, all polynomials have a multiplicative inverse.
- For $a(X) \in GF(2^m)$ and irreducible polynomial $m(X) \in GF(2^m)$ by the Chinese Remainder Theorem there exists polynomials $b(X), c(X) \in GF(2^m)$ such that

$$a(X)b(X) + m(X)c(X) = 1$$

or

$$a(X)b(X) \equiv 1 \pmod{m(X)}$$
$$\Rightarrow\ a^{-1}(X) = b(X) \pmod{m(X)}$$

Multiplicative Inverse

- We can now solve this equation with the Extended Euclidean Algorithm
- Consider

$$GF(2^3) = Z_2[X] \pmod{X^3 + X + 1}$$

- *Example* Find the inverse of $a(X) = X^2 + X + 1$ in $GF(2^3)$.

Step 1: Euclidean Algorithm:

$$X^3 + X + 1 = (X+1)(X^2 + X + 1) + (X)$$
$$X^2 + X + 1 = (X+1)(X) + 1$$

The last remainder is 1, which tells us that the greatest common divisor is 1
(cf. $X^3 + X + 1$ is irreducible.)

Multiplicative Inverse

- *Example (cont.)*

Step 2: Work backwards to write 1 as linear combination of the two polynomials.

$$X^3 + X + 1 = (X+1)(X^2+X+1) + (X) \Rightarrow \mathbf{X = X^3 + X + 1 + (X+1)(X^2+X+1)}$$
$$X^2 + X + 1 = (X+1)(X) + 1 \Rightarrow \mathbf{1 = (X^2+X+1) + (X+1)(X)}$$

$$\begin{aligned} 1 &= (X^2+X+1) + (X+1)(X) \\ &= (X^2+X+1) + (X+1)(X^3+X+1+(X+1)(X^2+X+1)) \\ &= (1+(X+1)^2)(X^2+X+1) + (X+1)(X^3+X+1) \\ &= (X^2)(X^2+X+1) + (X+1)(X^3+X+1) \end{aligned}$$

Hence,

$$(X^2)(X^2+X+1) \equiv 1 \pmod{X^3 + X + 1}$$

And

$$a^{-1}(X) \equiv X^2 \pmod{X^3 + X + 1}$$

$GF(2^m)$ and Rijndael

Basics of Rijndael (AES)

- In 2002, the National Institute of Standards and Technology (NIST) adopted the Advanced Encryption Standard (AES) also known as Rijndael.
- Currently the standard encryption algorithm that is designed to be used by Federal departments and agencies have information that requires encryption [NIST].
- Algorithm accepts a 128 bit sequence of plaintext information and cycles through four layers to produce the ciphertext which is also a 128 bit sequence of data.

$GF(2^m)$ and Rijndael

- The first step in the Rijndael algorithm is to group the 128 bit input into 16 bytes of 8 bits and arrange them into a 4×4 array of bytes.

Input bytes, State Array, Output bytes

$$\begin{pmatrix} in_1 & in_5 & in_9 & in_{13} \\ in_2 & in_6 & in_{10} & in_{14} \\ in_3 & in_7 & in_{11} & in_{15} \\ in_4 & in_8 & in_{12} & in_{16} \end{pmatrix} \Rightarrow \begin{pmatrix} s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \\ s_4 & s_8 & s_{12} & s_{16} \end{pmatrix} \Rightarrow \begin{pmatrix} out_1 & out_5 & out_9 & out_{13} \\ out_2 & out_6 & out_{10} & out_{14} \\ out_3 & out_7 & out_{11} & out_{15} \\ out_4 & out_8 & out_{12} & out_{16} \end{pmatrix}$$

- Input array is then manipulated by the 4-layer algorithm in 10, 12, or 14 rounds for key lengths of 128, 192, or 256 bits.

ByteSub (BS)

- A non-linear byte substitution routine that operates on each byte with bits $\{b_0, b_1 \ldots, b_7\}$ using the affine transformation

$$\begin{pmatrix} 1&0&0&0&1&1&1&1 \\ 1&1&0&0&0&1&1&1 \\ 1&1&1&0&0&0&1&1 \\ 1&1&1&1&0&0&0&1 \\ 1&1&1&1&1&0&0&0 \\ 0&1&1&1&1&1&0&0 \\ 0&0&1&1&1&1&1&0 \\ 0&0&0&1&1&1&1&1 \end{pmatrix} \cdot \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \end{pmatrix}.$$

- To simplify matters, we can compute this transformation on all possible bytes in $GF(2^8)$ place them in a look up table called an S-box.

ByteSub (BS)

S-box in hexadecimal format [NIST]

S-Box Values

S(rs)		s															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
R	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Consider {10011011} and use the first four digits {1001} = {9} which tell us to look in row 9, and the second four digits {1011} = {b} which gives us column b.
{10011011} = {9b} → {81} = {10000001}.

ShiftRow (SR)

- The ShiftRow (SR) layer offsets the bytes cyclically by 0, 1, 2, and 3 columns in rows 1, 2, 3, and 4 respectively.

$$\begin{pmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{pmatrix} \Rightarrow \begin{pmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,1} & s_{1,2} & s_{1,3} & s_{1,0} \\ s_{2,2} & s_{2,3} & s_{2,0} & s_{2,1} \\ s_{3,3} & s_{3,0} & s_{3,1} & s_{3,2} \end{pmatrix}$$

MixColumns (MC)

- In the MixColumn layer, each column becomes a 4-term polynomial in $GF(2^8)$ such as
$$\boldsymbol{b(x) = b_3X^3 + b_2X^2 + b_1X + b_0}$$
where b_0, b_1, b_2, and b_3 are bytes from the column of the shift matrix
- Multiply this polynomial modulo $(X^4 + 1)$ by
$$\boldsymbol{a(X) = \{0011\}X^3 + \{0001\}X^2 + \{0001\}X + \{0010\}}$$
- Note: $X^4 + 1$ is not irreducible, so an inverse is not guaranteed, but $a(x)$ does have an inverse:
$$\boldsymbol{a^{-1}(X) = \{1011\}X^3 + \{1101\}X^2 + \{1001\}X + \{1110\} \pmod{X^4 + 1}}$$

MixColumns (MC)

- This gives us
$$\begin{aligned} d_0 &= a_0 \cdot b_0 \oplus a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_3 \\ d_1 &= a_1 \cdot b_0 \oplus a_0 \cdot b_1 \oplus a_3 \cdot b_2 \oplus a_2 \cdot b_3 \\ d_2 &= a_2 \cdot b_0 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_2 \oplus a_3 \cdot b_3 \\ d_3 &= a_3 \cdot b_0 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3 \end{aligned}$$
- In Matrix form
$$\begin{pmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix}$$
This matrix sufficiently defines the MixColumn transformation when applied to each column $(b_0, b_1, b_2, b_3)^T$ of the shift matrix

MixColumns (MC)

- Let's analyze this product $a(X)b(X)$ with
$$a(X) = \sum_{i=0}^{3} a_i X^i, \qquad b(X) = \sum_{j=0}^{3} b_j X^j$$
- We note that when working (mod $X^4 + 1$):
$$X^i \equiv X^{i \bmod 4} (\bmod X^4 + 1)$$
- Multiplying and collecting like terms, we get
$$c(X) = a(X) \cdot b(X) = \sum_{n=i+j=0}^{6} c_n X^n = \sum_{n=i+j=0 \bmod 4}^{3 \bmod 4} d_n X^n$$
Where
$$d_n = \sum_{i+j=0 \bmod 4}^{3 \bmod 4} a_i \cdot b_j$$

AddRoundKey (ARK)

- XOR the shift matrix with the round key matrix as defined by the key schedule which is again defined by operations in $GF(2^8)$.
- The Rijndael system is designed to work with a key of 128, 192, or 256. (We'll use a 128-bit key)

<u>Key Schedule</u>

1. Arrange the 128-bit key into a 4×4 matrix of bytes
2. Add 40 columns to the matrix as follows:

AddRoundKey (ARK)

<u>Key Schedule (cont.)</u>

a) Designate the first 4 columns $W(0), W(1), W(2), W(3)$

b) For successive columns i,

If $i > 0 \bmod 4$, then
$$W(i) = W(i-4) \oplus W(i-1)$$
If $i \equiv 0 \bmod 4$, then
$$W(i) = W(i-4) \oplus T(W(i-1))$$

$T(W)$:Shift elements cyclically in column $W(i-1)$, Replace bytes with S-box values, compute
$$r(i) = 00000010^{(i-4)/4}, \text{ and}$$
$$T(W) = (e + r(i), f, g, h)$$

Encryption/Decryption

Rijndael Encryption Summary

- For the 128-bit key, we encryption includes

1. ARK with the 0th round key,
2. Nine rounds of BS, SR, MC, and ARK, using keys 1 to 9, and
3. Tenth round of BS, SR, and ARK using the 10th key.

Rijndael Decryption Summary

Each encryption layer is invertible! The reverse algorithm is:

1. ARK with 10th round key,
2. Nine rounds of IBS, ISR, IMC, IARK using keys 9 to 1
3. Tenth round of IBS, ISR, and ARK using the 10th key.

Decryption Layers

Inverse ByteSub (IBS)

- Apply the inverse affine transformation to each byte in the shift array and find the multiplicative inverse of the result in $GF(2^8)$.
- Again… we can use another look-up table.

Inverse ShiftRow (ISR)

- Shift rows to the right instead of the left by 0, 1, 2, and 3 entries, respectively
- Resulting in the byte-wise formula:

$$s'_{r,(c+shift(r,4)) \bmod 4} = s_{r,c}$$

Decryption Layers

Inverse MixColumn (IMC)

- Treat each column as a 4th –degree polynomial modulo $X^4 + 1$ in $GF(2^8)$.
- Compute the matrix product below column-by-column

$$\begin{pmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Where a_i entries are coefficients of

$$\boldsymbol{a^{-1}(X) = \{1011\}X^3 + \{1101\}X^2 + \{1001\}X + \{1110\} \pmod{X^4 + 1}}$$

Error Correction Codes

- Mistakes happen! When transmitting a cryptographic ciphertext, the corruption of even one bit can make a plaintext message unreadable.
- Digital data is susceptible to errors from bit reversal due to “noise”.
- Error correction codes can identify a bit or bits that have been altered so they can be returned to their original state.

Hamming Codes

- Finite fields are the key to many useful Error correction codes.
- Types of Hamming Codes: Linear and Cyclic
- $[n, k]$ block code: encodes a k-bit information word to an n-bit codeword.
- Step 1: multiply the k-bit information word by <u>generating</u> matrix.
- Step 2: (after transmision) multiply the n-bit codeword by <u>parity check</u> matrix.

Linear Codes

- We will work with $GF(2^3)$ to create a hamming matrix. [RT1]
- We first need a primitive polynomial: monic irreducible polynomial whose roots are primitive elements.
- We found irreducible polynomial

$$P(X) = X^3 + X^2 + 1$$

- Is it Primitive in $GF(2^3)$? Let’s Check
- If a is a root, then

$$P(a) = a^3 + a^2 + 1 = 0 \;\Rightarrow\; a^3 = a^2 + 1$$

Linear Codes

$a^0 = a^0$	$= 1$		$= 001 = 1$
$a^1 = a^0 \times a$	$= 1 \times a$	$= a$	$= 010 = 2$
$a^2 = a^1 \times a$	$= a \times a$	$= a^2$	$= 100 = 4$
$a^3 = a^2 + 1$	$= a^2 + 1$	$= a^2 + 1$	$= 101 = 5$
$a^4 = a^3 \times a$	$= (a^2 + 1) \times a$	$= a^3 + a$ $= a^2 + a + 1$	$= 111 = 7$
$a^5 = a^4 \times a$	$= (a^2 + a + 1) \times a$	$= a^3 + a^2 + a$ $= a^2 + 1 + a^2 + a$ $= a + 1$	$= 011 = 3$
$a^6 = a^5 \times a$	$= (a + 1) \times a$	$= a^2 + a$	$= 110 = 6$
$a^7 = a^6 \times a$	$= (a^2 + a) \times a$	$= a^3 + a^2 = a^2 + 1 + a^2 = 1$	$= 001 = 1$

Parity Check Matrix

- We see from the power $a^7 = 1$ that we have the cyclic group $GF^*(2^3)$ working $(\text{mod } X^3 + X^2 + 1)$
- **Theorem [TW]:** If $G = [I_k, P]$ is the generating matrix for a code C, then $H = [-P^T, I_{n-k}]$ is the parity check matrix for C.
- Constructing the parity check matrix [RT1]: Compile the remainders in the table into the matrix

$$[a^6 \quad a^5 \quad a^4 \quad a^3 \quad a^2 \quad a^1 \quad a^0] =$$

$$H = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}^T$$

Generating Matrix

- Using the theorem [TW] stated previously, the generating matrix is:

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

- ***Example*** Suppose we begin with a plaintext word, $p = 1010$, We will encode it with G, alter one bit, then use the parity check matrix H to identify the error.

Hamming Example

Example (cont)

Step 1. Compute code word $c = pG$

$$c = pG = (1 \quad 0 \quad 1 \quad 0) \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

$$= (1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1)$$

- Observe: codeword contains the plaintext word followed by three check bits 001.
- Now, we will alter the 4th bit (to simulate an error):

$$c' = (1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1)$$

Hamming Example

Example (cont)

$$c' = (1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1)$$

Step 2. Compute the check bit product $c'H$

$$c'H = (1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1) \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}^T$$

$$= (1 \quad 0 \quad 1)$$

- Note: unaltered codewords produce $cH = \mathbf{0}$.
- $c'H = (1 \quad 0 \quad 1)$ is the 4th column of H.
- Hence 4th bit was changed.

Hamming Example

Example (cont)

Step 3: Correct bit and check

$$c = (1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1)$$

$$cH = (1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1) \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}^T$$

$$= (0 \quad 0 \quad 0)$$

Finite Elements and P.D.E.s

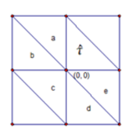
- The applications of Partial Differential Equations (PDEs) are
- Many analytic techniques are available for solving linear PDEs in standard forms:
- Wave equation:
  $$-a^2u_{xx} + u_{tt} + cu = F(x,t), \qquad k > 0$$
- Poisson/Laplace equation:
  $$a^2u_{xx} + u_{tt} + cu = g(x,t), \qquad a > 0$$
- Heat equation:
  $$-ku_{xx} + u_t + cu = h(x,t), \qquad a > 0$$

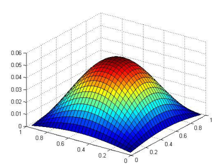
Numerical Methods

- In many applications, the equations become too complex for known analytical methods.
- In this case, we must use numerical methods of approximating solutions to PDEs.
- Difference methods-
  - approximate derivatives by calculating differences over increasingly small intervals that converge to the analytical solution
- Other useful methods:
  - Crank-Nicolson method and Rayleigh-Ritz method

Finite Element Method

- A given region is divided into a finite number of geometric sub-regions, called the finite elements.
- Use a set of basis functions from a chosen function space to extrapolate the values of the solution for each finite element using initial values and boundary values





Finite Element Method

The key steps:

- Define our finite element space V_h and the nature and parameters of the functions v in V_h.
- Compute the local stiffness matrix and the coefficients of the local basis functions.
- Compute the values of the global nodes and map the local nodes to global nodes.
- Compute the global stiffness matrix, S, (coefficients of the system which we need to solve.)
- Compute the values of the vector $b = (b_i)$, where $b_i = \int_{\Omega} f(x,y)\phi_i(x,y)dx$
- Finally, we will solve the matrix equation $Sx = $ **b.**

FEM and Poisson's Equation

Let's consider the Poisson problem:

$$-\Delta u = f \quad in \quad \Omega, \qquad \Omega = [0,1]^2$$
$$u = 0 \quad on \quad \partial\Omega$$

We will implement FEM with MATLAB algorithms.

- http://knightmath.com/tamu/poisson/

2^k Factorial Design

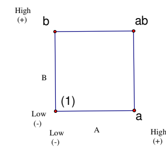
- Experiments are an important tool for all areas of science and engineering.
- We must carefully consider the *design* of the experiment.
- Often a result is affected by multiple factors.
- It is useful to perform a *factorial experiment*, which is performed at all factor levels.

2^k *Factorial Design*

- With k factors that can be controlled, we can use a 2^k factorial design.
- Analyze the effects of the individual factors as well as the joint effect of the factors on the response.
- Quantitative *or* qualitative responses studied at <u>only two levels</u> for each factor.
- Called a 2^k factorial design since the experiment requires 2^k observations. [MR]

2^k *Factorial Design*

- 2^2 factorial design $\Rightarrow$ two factors (A and B)
- Observe these factors at two levels, low(-) and high(+),
- Requires $2^2 = 4$ observations as shown in the geometric model
- $(1), a, b,$ and ab represent the total of all n observations taken at these levels.
- Design addresses all possible factors and interactions.



Treat ment	A	B
(1)	-	-
a	+	-
b	-	+
ab	+	+

2^k *Factorial Design*

- In a 2^k factorial design, the combinations of the (+) and (-) symbols mirror the binary representation of the polynomials in $GF(2^k)$.
- Testing of each of the factors at many levels is often unnecessary.
- The factorial model gives us a good picture of which factors are significant by testing them at only two levels each.

Clinical Samples

<u>*Example*</u>:
An article in *Analytica Chimica Acta* examined four parameters that affect the sensitivity and detection of the analytical instruments used to measure clinical samples. They optimized the sensor function using EBC samples spiked with acetone, a known clinical biomarker in breath. The following table shows the results for a single replicate of a 2^4 factorial experiment for one of the outputs, the average amplitude of acetone peak over three repetitions.

Clinical Samples

Configuration	A	B	C	D	Yield
1	+	+	+	+	0.12
2	+	+	+	-	0.1193
3	+	+	-	+	0.1196
4	+	+	-	-	0.1192
5	+	-	+	+	0.1186
6	+	-	+	-	0.1188
7	+	-	-	+	0.1191
8	+	-	-	-	0.1186
9	-	+	+	+	0.121
10	-	+	+	-	0.1195
11	-	+	-	+	0.1196
12	-	+	-	-	0.1191
13	-	-	+	+	0.1192
14	-	-	+	-	0.1194
15	-	-	-	+	0.1188
16	-	-	-	-	0.1188

A: RF voltage of the DMS Sensor (1200 or 1400V)

B: Nitrogen carrier gas flow rate (250 or 500)

C: Solid phase microextraction filter type (polyacrylate or PDMS-DVB)

D: GC cooling profile (cryogenic and noncryogenic)

Clinical Samples

- **Data:** A 2^4 factorial experiment on clinical samples
  Objective: Factor analysis and interaction of factor using an effects model
  Hypotheses: We consider the null hypotheses below with a confidence level of 95%.

- Main Effects – $H_0: (\alpha)_i = 0$
  2-way Interaction Effects - $H_0: (\alpha\beta)_{ij} = 0$
  3-way Interaction Effects – $H_0: (\alpha\beta\gamma)_{ijk} = 0$
  (We will ignore 4-way interactions since they are highly unlikely to be significant.)

Clinical Samples

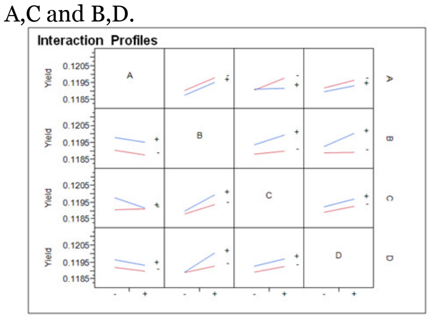
- Significant main effects: B, C, and D (p-value<.05)
- Significant two-way interactions: A*C and B*D.

Effect Tests

Source	Nparm	DF	Sum of Squares	F Ratio	Prob > F
A	1	1	3.025e-7	121.0000	0.0577
B	1	1	0.00000225	900.0000	0.0212*
C	1	1	5.625e-7	225.0000	0.0424*
D	1	1	0.00000064	256.0000	0.0397*
A*B	1	1	4.8148e-35	0.0000	1.0000
A*C	1	1	4.225e-7	169.0000	0.0489*
A*D	1	1	0.00000001	4.0000	0.2952
B*C	1	1	0.00000016	64.0000	0.0792
B*D	1	1	5.625e-7	225.0000	0.0424*
C*D	1	1	0.00000001	4.0000	0.2952
A*B*C	1	1	0	0.0000	1.0000
A*B*D	1	1	1.225e-7	49.0000	0.0903
A*C*D	1	1	0.00000009	36.0000	0.1051
B*C*D	1	1	3.025e-7	121.0000	0.0577

Clinical Samples

- interaction profile plots show interactions of A,C and B,D.



Clinical Samples

- To analyze the fit of the effects model, we look at the ANOVA table and see that the p-value for the model fit is .0628 which is too high.

Analysis of Variance (Factors A, B, C, and D)

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	14	5.435e-6	3.8821e-7	155.2857
Error	1	2.5e-9	2.5e-9	Prob > F
C. Total	15	5.4375e-6		0.0628

Summary of Fit

RSquare	0.99954
RSquare Adj	0.993103
Root Mean Square Error	0.00005
Mean of Response	0.119288
Observations (or Sum Wgts)	16

Clinical Samples

- Since A is not a main effect, we will remove this factor and recalculate the model. When we do this, we get a p-value of .0150. This is an acceptable value for our goodness of fit.

Analysis of Variance (Factors B, C, and D)

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	7	4.4875e-6	6.4107e-7	5.3985
Error	8	0.00000095	1.1875e-7	Prob > F
C. Total	15	5.4375e-6		0.0150*

Summary of Fit

RSquare	0.825287
RSquare Adj	0.672414
Root Mean Square Error	0.000345
Mean of Response	0.119288
Observations (or Sum Wgts)	16

Conclusion

- Many real-life applications of mathematics require continuousness and infinite considerations.
- However, considering finite sets can prove quite effective in understanding and controlling results.
- In all situations, finite sets must be designed with much care and forethought to produce useful results.
- If this is done, we can *begin* to understand the infinite from our finite perspective.

[TW] Trappe, Wade, and Washington, Lawrence C. *Introduction to Cryptography with Code Theory. 2nd Edition.* Prentice Hall. 2006.

[MR] Montgomery, Douglas C. and Runger, George C. *Applied Statistics and Probability for Engineers. 5th edition.* John Wiley and Sons. 2011.

[RT1] Tervo, Richard. *Error Control Codes from Galois Fields.* Course notes for EE4253 Digital Communications. University of New Brunswick.
http://www.ee.unb.ca/cgi-bin/tervo/galois.pl

[RT2] Tervo, Richard. *The Hamming Code Revisited – A Matrix Approach.* Course notes for EE4253 Digital Communications. University of New Brunswick.
http://www.ee.unb.ca/tervo/ee4253/hamming2.shtml

[NW] Wagner, Niel R., The Laws of Cryptography; The Finite Field $GF(2^8)$
http://www.cs.utsa.edu/~wagner/laws/FFM.html
http://www.cs.utsa.edu/~wagner/lawsbookcolor/laws.pdf

[NIST] *Specification for the Advanced Encryption Standard (AES).* National Institute of Standards and Technology. November 26, 2001.
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[HM] Hansen, Tom and Mullen, G.L. *Primitive Polynomials Over Finite Fields.* Mathematics of Computation. American Mathematical Society. 1992.
http://www.ams.org/journals/mcom/1992-59-200/S0025-5718-1992-1134730-7/S0025-5718-1992-1134730-7.pdf

[MK] Kyuregyan, Melsik K. *Iterated constructions of irreducible polynomials over finite fields with linearly independent roots.* Science Direct. 2003.
http://web.mit.edu/minilek/Public/irreducible.pdf