Lecture 20

Golomb Decoding:

http://www.ece.uah.edu/~dwpan/course/ee614/code/golomb_deco.m

```
unction n_rec = golomb_deco (codeword, m)
len = length(codeword);
% Count the number of 1's followed by the first 0
q = 0;
for i = 1: len
   if codeword(i) == 1
      q = q + 1;
   else
      ptr = i;   % first 0
      break;
   end
end
if (m == 1)
   n_rec = q;   % special case for m = 1
else
   A = ceil(log2(m));
   B = floor(log2(m));
   bcode = codeword((ptr+1): (ptr + B));
   r = bi2de(bcode,'left-msb');
   if r < (2^A - m)
      ptr = ptr + B;
   else
      % r is A-bit representation of (r + (2^A - m))
      bcode = codeword((ptr+1): (ptr + A));
      r = bi2de(bcode,'left-msb') - (2^A - m);
      ptr = ptr + A;
   end
   n_rec = q * m + r;
end
if ~isequal(ptr, len)
   error('Error: More than one codeword detected!');
end
```

If m is a power of two, then

$A = B = \log_2 m$

$r \in (0, 1, \cdots, 2^A - m - 1), (2^A - m, 2^A - m + 1, \cdots, m - 1)$

Same implementation for both cases (m is a power of 2, or m is NOT a power of 2).

- Skip the next bit ('0').
- First assume that r is represented by the next B-bit binary code, then check if $0 \leq r \leq 2^A - m - 1$.

  If "Yes", then the assumption is valid;
  If "No", decode the next A-bit binary representation:

  $r = (\quad) - \underbrace{\text{offset}}_{2^A - m}$

http://www.ece.uah.edu/~dwpan/course/ee614/code/lossless_test.m

```
for i = 1: total
   x = golomb_enco(n(i), m(i));
   y = golomb_deco(x, m(i));
   if ~isequal(n(i), y)
……
```

$m = 14$			
n	Codeword	n	Codeword
0	0000	24	101100
1	0001	25	101101
2	00100	26	101110
3	00101	27	101111
4	00110	28	110000
5	00111	29	110001
6	01000	30	1100100
7	01001	31	1100101
8	01010	32	1100110
9	01011	33	1100111
10	01100	34	1101000
11	01101	35	1101001
12	01110	36	1101010
13	01111	37	1101011
14	10000	38	1101100
15	10001	39	1101101
16	100100	40	1101110
17	100101	41	1101111
18	100110	42	1110000
19	100111	43	1110001
20	101000	44	11100100
21	101001	45	11100101
22	101010	46	11100110
23	101011	47	11100111

	$m = 1$	
n	$G(n)$	Codeword
0	1/2	0
1	1/4	10
2	1/8	110
3	1/16	1110
4	1/32	11110
5	1/64	111110
6	1/128	1111110
7	1/256	11111110
8	1/512	111111110
9	1/1024	1111111110
10	1/2048	11111111110

```
>> golomb_deco([1 1 1 1 1 1 0], 1)
ans =
   6
```

```
>> golomb_deco([1 1 1 0 0 1 0 0], 14)
ans =
   44
```

RV X with **Geometric** Distribution
X: takes non-negative integer values (n)

$G(n) = P(X=n) = p^n(1-p)$, where $0<p<1$: given parameter

Next, determine the mean E[X]:

$G(n) = p^n(1-p) = P[X=n]$

$$E[X] = \sum_{n=0}^{\infty} n \cdot P[X=n] = \boxed{\frac{p}{1-p}} \Rightarrow 1+E[X] = 1+\frac{p}{1-p} = \frac{1}{1-p} \Rightarrow 1-p = \frac{1}{1+E[X]}$$

$$\Rightarrow p = 1 - \frac{1}{1+E[X]} = \frac{E[X]}{1+E[X]}$$

geornd

Geometric random numbers

The geometric distribution is useful to model the **number of failures** before one success in a series of independent trials, where each trial results in either success or failure, and the **probability of success** in any individual trial is the constant p.

```
>> X = geornd(1 - 0.9, 1, 1000000);
>> mean(X)
ans =
   8.9859
```

```
>> p = 0.9;
>> p/(1-p)
ans =
   9.0000
```

$$p = 1 - \frac{1}{1+E[X]} = \frac{E[X]}{1+E[X]}$$

$$p^m = \frac{1}{2} \Rightarrow m \log_2 p = -1 \Rightarrow m = \frac{-1}{\log_2 p} = \left\lceil \frac{-1}{\log_2 \frac{E[X]}{1+E[X]}} \right\rceil \leftarrow \text{ceiling}$$

Since m is an integer.

The procedure of implementing a predictive coding method called differential pulse-code modulation (DPCM) is explained using an example below:

Suppose I = $\begin{bmatrix} 1 & 3 & 2 \\ 6 & 8 & 7 \end{bmatrix}$ is a 2 × 3 image. We first convert the 2D image into a 1D vector as

(e) Determine the Golomb coding parameter m based on the converted residue sequence using $m = \left\lceil -\frac{1}{\log_2\left(\frac{E}{1+E}\right)} \right\rceil$, where ⌈ ⌉ is the ceiling function, and E is the sample mean of the converted residue sequence. What is the value of m?

```
>> ceil(-1/log2(mean(X)/(1+mean(X))))
ans =
    7
```

- Next set of HW problems