

Digital Communication Systems

ECS 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

4. Mutual Information and Channel Capacity



Office Hours:

BKD, 4th floor of Sirindhralai building

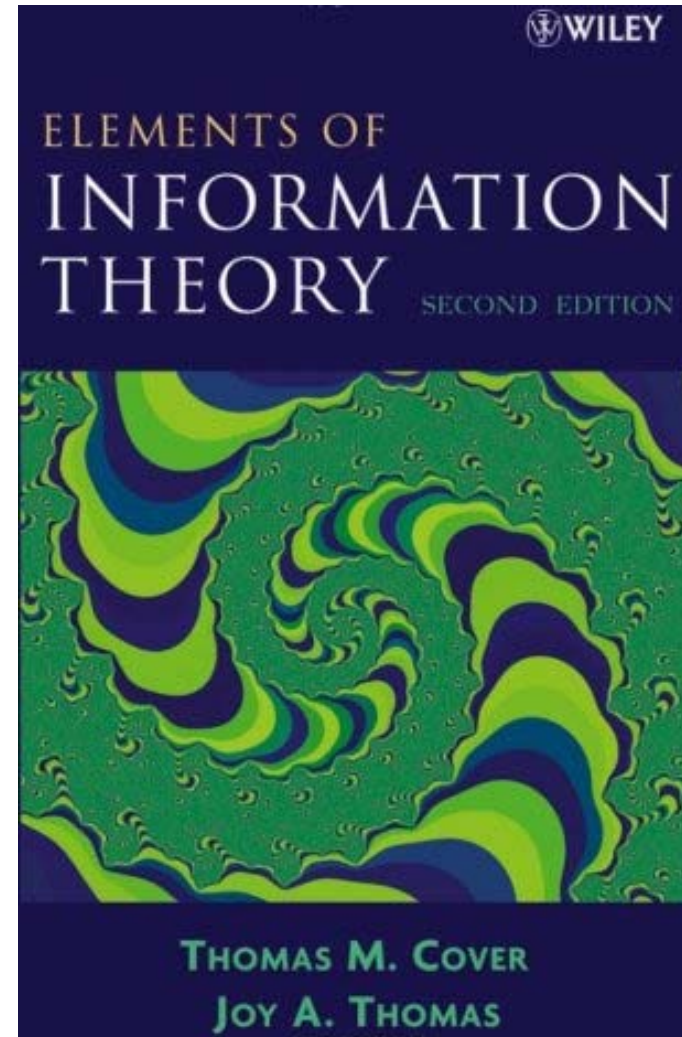
Monday 14:00-16:00

Thursday 10:30-11:30

Friday 12:00-13:00

Reference for this chapter

- Elements of Information Theory
- By Thomas M. **Cover** and Joy A. **Thomas**
- 2nd Edition (Wiley)
- Chapters 2, 7, and 8
- 1st Edition available at SIIT library: Q360 C68 1991



MATLAB

```
function H = entropy2s(p)
% ENTROPY2 accepts probability mass function
% as a row vector, calculate the corresponding
% entropy in bits.
p=p(find(abs(sort(p)-1)>1e-8)); % Eliminate 1
p=p(find(abs(p)>1e-8)); % Eliminate 0
if length(p)==0
    H = 0;
else
    H = simplify(-sum(p.*log(p))/log(sym(2)));
end
```

```
function I = informations(p,Q)
X = length(p);
q = p*Q;
HY = entropy2s(q);
temp = [];
for i = 1:X
    temp = [temp entropy2s(Q(i,:))];
end
HYgX = sum(p.*temp);
I = HY-HYgX;
```

Digital Communication Systems

ECS 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

Operational Channel Capacity

Digital Communication Systems

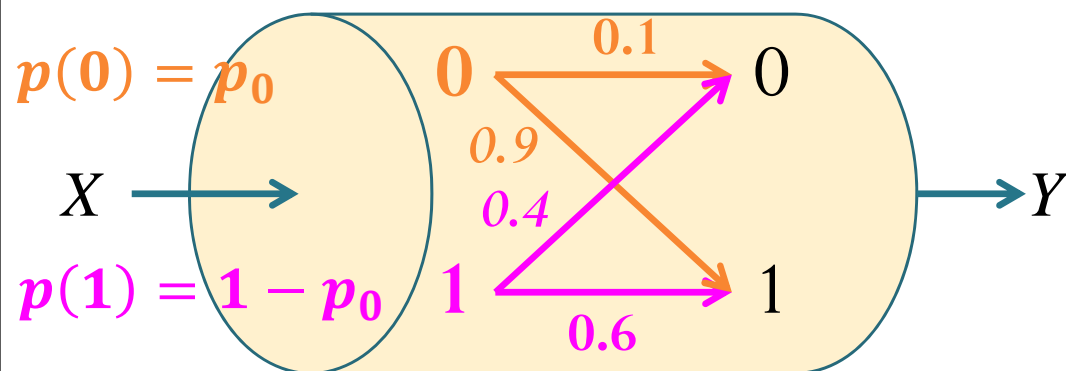
ECS 452

Asst. Prof. Dr. Prapun Suksompong

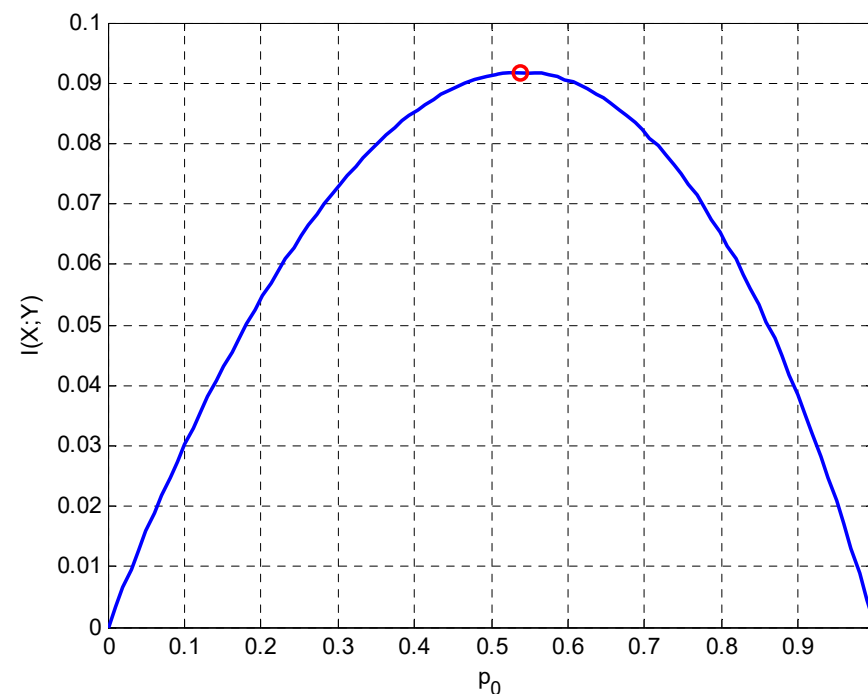
prapun@siit.tu.ac.th

Information Channel Capacity

Capacity calculation for BAC

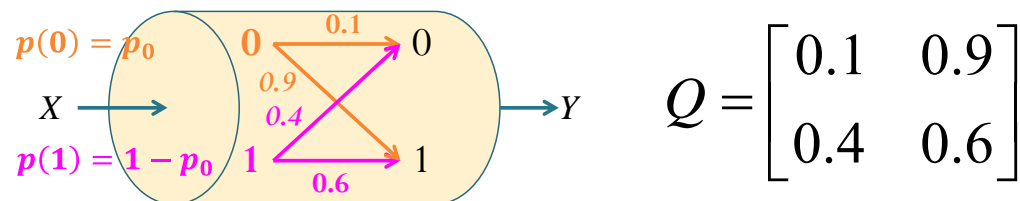


$$Q = \begin{bmatrix} 0.1 & 0.9 \\ 0.4 & 0.6 \end{bmatrix}$$



Capacity of 0.0918 bits is achieved by $\underline{p} = [0.5380, 0.4620]$

Capacity calculation for BAC



```
close all; clear all;
syms p0
p = [p0 1-p0];
Q = [1 9; 4 6]/sym(10);
```

```
I = simplify(informations(p,Q))
```

```
p0o = simplify(solve(diff(I)==0))
```

```
po = eval([p0o 1-p0o])
```

```
C = simplify(subs(I,p0,p0o))
```

```
eval(C)
```

```
>> Capacity_Ex_BAC
```

```
I =
```

```
(log(2/5 - (3*p0)/10)*((3*p0)/10 - 2/5) - log((3*p0)/10 + 3/5)*((3*p0)/10 + 3/5))/log(2) + (log((5*2^(3/5)*3^(2/5))/6)*(p0 - 1))/log(2) + (p0*log((3*3^(4/5))/10))/log(2)
```

```
p0o =
```

```
(27648*2^(1/3))/109565 - (69984*2^(2/3))/109565 + 135164/109565
```

```
po =
```

```
0.5376 0.4624
```

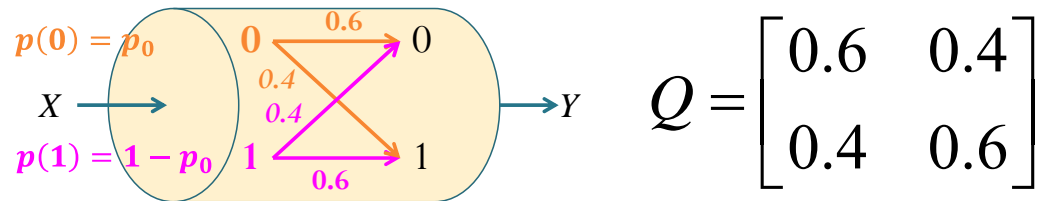
```
C =
```

```
(log((3*3^(4/5))/10)*((27648*2^(1/3))/109565 - (69984*2^(2/3))/109565 + 135164/109565))/log(2) - (log((104976*2^(2/3))/547825 - (41472*2^(1/3))/547825 + 16384/547825)*((104976*2^(2/3))/547825 - (41472*2^(1/3))/547825 + 16384/547825) + log((41472*2^(1/3))/547825 - (104976*2^(2/3))/547825 + 531441/547825)*((41472*2^(1/3))/547825 - (104976*2^(2/3))/547825 + 531441/547825))/log(2) + (log((5*2^(3/5)*3^(2/5))/6)*((27648*2^(1/3))/109565 - (69984*2^(2/3))/109565 + 25599/109565))/log(2)
```

```
ans =
```

```
0.0918
```

Same procedure applied to BSC



```
close all; clear all;
syms p0
p = [p0 1-p0];
Q = [6 4; 4 6]/sym(10);
```

```
I = simplify(informations(p,Q))
```

```
p0o = simplify(solve(diff(I)==0))
```

```
po = eval([p0o 1-p0o])
```

```
C = simplify(subs(I,p0,p0o))
```

```
eval(C)
```

```
>> Capacity_Ex_BSC
```

```
I =
```

```
(log((5*2^(3/5)*3^(2/5))/6)*(p0 - 1))/log(2) -
(p0*log((5*2^(3/5)*3^(2/5))/6))/log(2) - (log(p0/5 +
2/5)*(p0/5 + 2/5) - log(3/5 - p0/5)*(p0/5 -
3/5))/log(2)
```

```
p0o =
```

```
1/2
```

```
po =
```

```
0.5000 0.5000
```

```
C =
```

```
log((2*2^(2/5)*3^(3/5))/5)/log(2)
```

```
ans =
```

```
0.0290
```

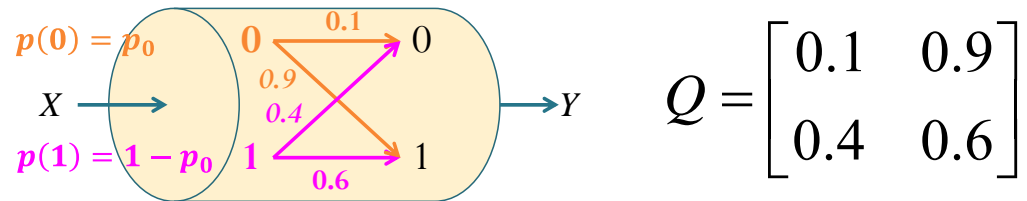

Blahut–Arimoto algorithm

```
function [ps C] = capacity_blahut(Q)
% Input:      Q = channel transition probability matrix
% Output:     C = channel capacity
%            ps = row vector containing pmf that achieves capacity

t1 = 1e-8; % tolerance (for the stopping condition)
n = 1000; % max number of iterations (in case the stopping condition
          % is "never" reached)
nx = size(Q,1); pT = ones(1,nx)/nx; % First, guess uniform X.
for k = 1:n
    qT = pT*Q;
    % Eliminate the case with 0
    % Column-division by qT
    temp = Q.*(ones(nx,1)*(1./qT));
    %Eliminate the case of 0/0
    l2 = log2(temp);
    l2(find(isnan(l2) | (l2==-inf) | (l2==inf)))=0;
    logc = (sum(Q.*(l2),2))';
    CT = 2.^(logc);
    A = log2(sum(pT.*CT)); B = log2(max(CT));
    if((B-A)<t1)
        break
    end
    % For the next loop
    pT = pT.*CT; % un-normalized
    pT = pT/sum(pT); % normalized
    if(k == n)
        fprintf('\nNot converge within n loops\n')
    end
end
ps = pT;
C = (A+B)/2;
```

[capacity_blahut.m]

Capacity calculation for BAC: a revisit



```
close all; clear all;  
  
Q = [1 9; 4 6]/10;  
  
[ps C] = capacity_blahut(Q)
```

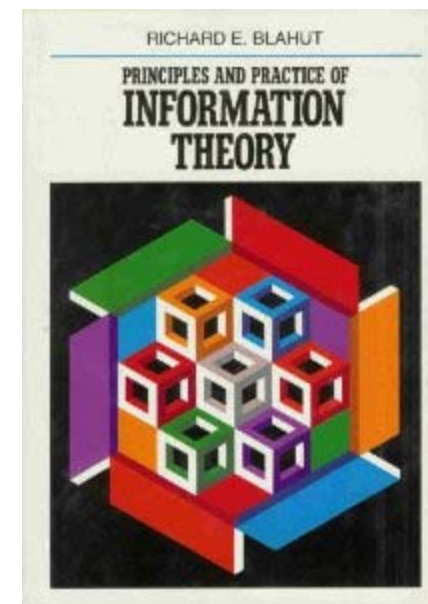
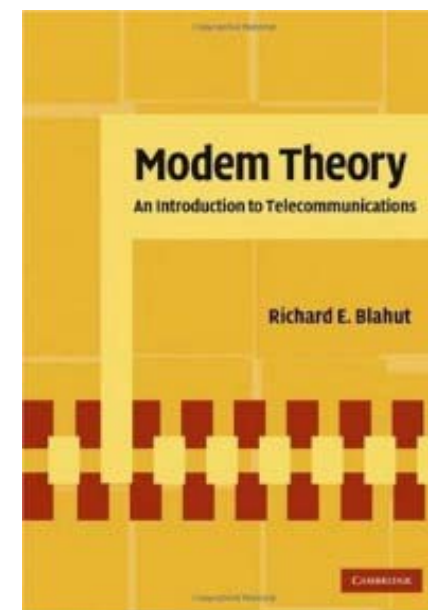
```
>> Capacity_Ex_BAC_blahut  
ps =  
    0.5376    0.4624  
C =  
    0.0918
```

Berger plaque



Richard Blahut

- Former chair of the Electrical and Computer Engineering Department at the University of Illinois at Urbana-Champaign
- Best known for **Blahut–Arimoto algorithm** (Iterative Calculation of C)



Raymond Yeung

- BS, MEng and PhD degrees in electrical engineering from **Cornell** University in 1984, 1985, and 1988, respectively.

