ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΘΕΩΡΙΑ ΠΛΗΡΟΦΟΡΙΑΣ

6η ΣΕΙΡΑ ΓΡΑΠΤΩΝ ΑΣΚΗΣΕΩΝ

Ειρήνη Δόντη

ΑΜ: 03119839

9ο εξάμηνο

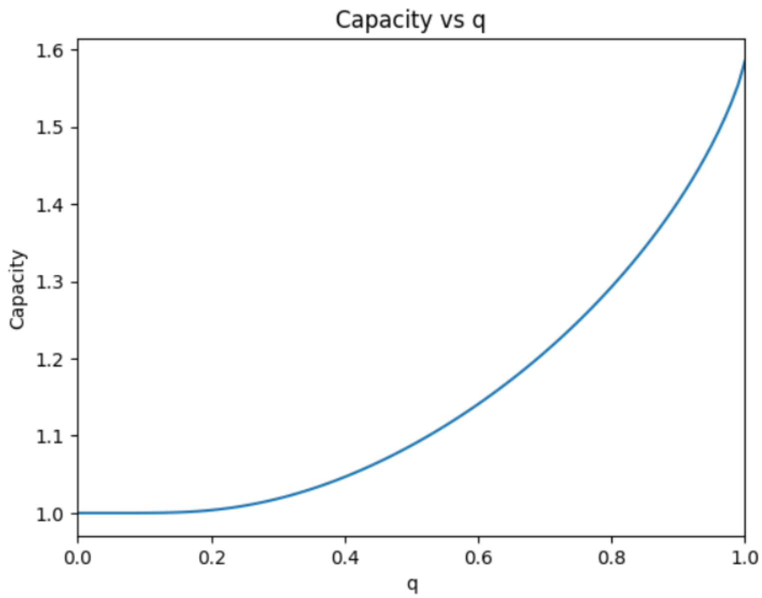
Αθήνα 2023-2024

Για την επίλυση των μαθηματικών σχέσεων της εκάστοτε άσκησης χρησιμοποιούμε κώδικα σε Python.

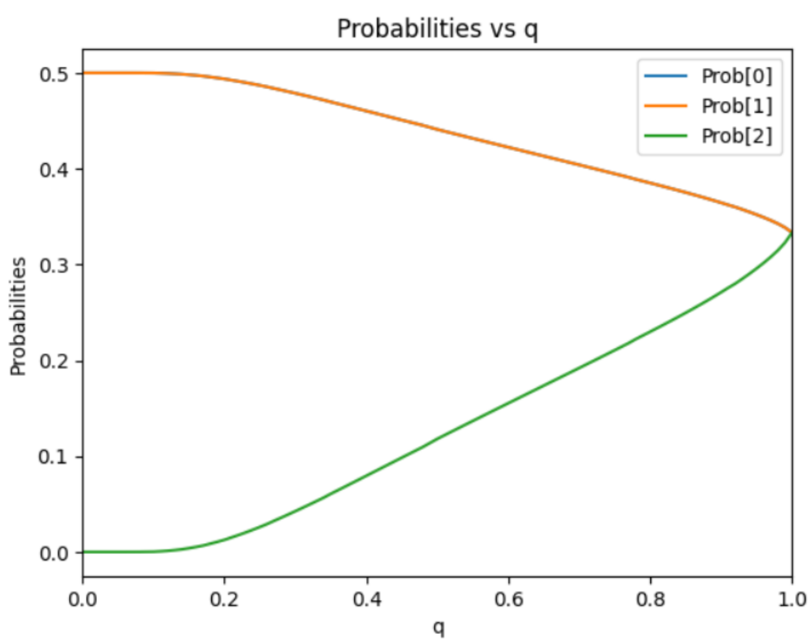
<u>Άσκηση 1</u>

1. Η χωρητικότητα του καναλιού προκύπτει ως η μέγιστη αμοιβαία πληροφορία I(X;Y), λαμβάνοντας υπόψιν όλες τις πιθανές κατανομές της p(x), δηλαδή C = $\max_{p(x)}$I(X;Y). Πρέπει να εξετάσουμε για ποια κατανομή της p(x) μεγιστοποιείται η I(X;Y). Έστω ότι p(x) είναι (p_0, p_1, p_2) και H(Y) = $-\sum_y p(y)\log(p(y)) = -\sum_y\{\sum_x p(x)p(y|x)\log\big(\sum_x p(x)p(y|x)\big)\}$ =
   -$(1p_0+ 0p_1+\frac{1}{2}(1-q)p_2)\, log_2(1p_0+ 0p_1+\frac{1}{2}(1-q)p_2)$
   -$(0p_0+ 1p_1+\frac{1}{2}(1-q)p_2)\, log_2(0p_0+ 1p_1+\frac{1}{2}(1-q)p_2)$
   -$(0p_0 + 0p_1 + qp_2)\, log_2(0p_0 + 0p_1 + qp_2)$
   Επίσης, ισχύει ότι H(Y|X) = $-\sum_x p(x) \sum_y p(y|x)\log(p(y|x))$ = p_0H(r_1) + p_1H(r_2) + p_2H(r_3), όπου με H(r_i) την εντροπία της i-οστής γραμμής, αριθμός υπολογίσιμος και ίδιος με την περίπτωση των ισοπίθανων συμβόλων εισόδου. Καλούμαστε να λύσουμε το πρόβλημα γραμμικού προγραμματισμού.
   max H(Y) – H(Y|X), όπου $p_0 + p_1 + p_2 = 1$ και $0 \le p_0, p_1, p_2 \le 1$.
   Για *q = 0*: Με τη βοήθεια του μαθηματικού εργαλείου, το μέγιστο επιτυγχάνεται στην κατανομή (0.5, 0.5, 0) και η ακριβής χωρητικότητα υπολογίζεται ως C = 1.0000 bits.
   Για $q = \frac{1}{2}$: Με τη βοήθεια του μαθηματικού εργαλείου, το μέγιστο επιτυγχάνεται στην κατανομή (0.4410, 0.4410, 0.1180) και η ακριβής χωρητικότητα υπολογίζεται ως C = 1.0875 bits.
   Για *q = 1*: Με τη βοήθεια του μαθηματικού εργαλείου, το μέγιστο επιτυγχάνεται στην κατανομή (0.3333, 0.3333, 0.3333) και η ακριβής χωρητικότητα υπολογίζεται ως C = 1.5850 bits.

2. Εφαρμόζοντας την παραπάνω διαδικασία για τιμές q από 0 ως 1, δημιουργούμε το διάγραμμα της χωρητικότητας καναλιού, όπως φαίνεται παρακάτω:



Επίσης, δημιουργούμε το διάγραμμα με τις τρεις τιμές $p_i = \Pr\{X = i\}$, i = 0,1,2 για q από 0 ως 1, όπως φαίνεται παρακάτω:



Άσκηση 2

1. Χωρίς βλάβη της γενικότητας θεωρούμε ότι τα σύμβολα που μεταδίδονται είναι 0, 1, 2, 3. Επίσης, γνωρίζουμε ότι ισχύει I(X;Y) = H(Y) – H(Y|X). Όταν τα σύμβολα εισόδου είναι ισοπίθανα, τότε Pr{X = x} = $\frac{1}{|X|}$ = $\frac{1}{4}$, x = 0, 1, 2, 3.

   Για να υπολογίσουμε το H(Y), χρειαζόμαστε την κατανομή της p(y), την οποία μπορούμε να υπολογίσουμε ως p(y) = $\sum_x p(x)p(y|x)$, επειδή όμως τα σύμβολα εισόδου είναι ισοπίθανα ισχύει ότι p(y) = $\frac{1}{4}\sum_x p(y|x)$ (όπου $\sum_x p(y|x)$ το άθροισμα της y στήλης). Οπότε, η κατανομή της p(y) = (0.245, 0.265, 0.250, 0.240). Για τον υπολογισμό της H(X|Y) = -$\sum_x p(x) \sum_y p(y|x)\log(p(y|x))$ = $\frac{1}{4}$(H(r_1) + H(r_2) + H(r_3) + H(r_4)), με H(r_i) την εντροπία της i-οστής γραμμής. Οπότε, προκύπτει ότι I(X;Y) = 1.3162 με τη βοήθεια του μαθηματικού εργαλείου. Επίσης, η χωρητικότητα καναλιού, στην προκειμένη περίπτωση είναι C= $\max_{p(x)}$I(X;Y) = (1+0.2)1.3162 = 1.5794, λόγω της πιθανότητας σφάλματος από 15 ως 20%. Ένα κάτω φράγμα για τη χωρητικότητα καναλιού είναι 1.0530, γιατί υπάρχει πιθανότητα σφάλματος από 15 ως 20% I(X;Y) ≥ (1-0.2)1.3162 ή I(X;Y) ≥ 1.0530.

2. Η γραμμή που ελαχιστοποιεί τη μέση τετραγωνική απόσταση, είναι η τρίτη γραμμή του πίνακα. Στην περίπτωση αυτή, ο πίνακας είναι συμμετρικός και επομένως η χωρητικότητα υπολογίζεται ως: C = log|Y| - H(r) = 0.9639 bits.

3. Ανάμεσα στις προσεγγίσεις της ακριβούς χωρητικότητας 1 και 2, θα προτιμούσαμε την πρώτη προσέγγιση, καθώς έχει τη μεγαλύτερη χωρητικότητα καναλιού.

4. Η χωρητικότητα του καναλιού προκύπτει ως η μέγιστη αμοιβαία πληροφορία I(X;Y), λαμβάνοντας υπόψιν όλες τις πιθανές κατανομές της p(x), δηλαδή C = $\max_{p(x)}$I(X;Y). Πρέπει να εξετάσουμε για ποια κατανομή της p(x) μεγιστοποιείται

η I(X;Y). Έστω ότι p(x) είναι (p_0, p_1, p_2, p_3) και H(Y) = $-\sum_y p(y)\log(p(y))$ =

$-\sum_y \{\sum_x p(x)p(y|x)\log\big(\sum_x p(x)p(y|x)\big)\}$ =

-(0.80p_0+ 0.05p_1+ 0.08p_2+0.05p_3) log_2(0.80p_0+ 0.05p_1+ 0.08p_2+0.05p_3)

-(0.10p_0+ 0.85p_1+ 0.06p_2+0.05p_3) log_2(0.10p_0+ 0.85p_1+ 0.06p_2+0.05p_3)

-(0.10p_0+ 0.05p_1+ 0.80p_2+0.05p_3) log_2(0.10p_0+ 0.05p_1+ 0.80p_2+0.05p_3)

-(0.00p_0+ 0.05p_1+ 0.06p_2+0.85p_3) log_2(0.00p_0+ 0.05p_1+ 0.06p_2+0.85p_3)

Επίσης, ισχύει ότι H(Y|X) = $-\sum_x p(x) \sum_y p(y|x)\log(p(y|x))$ = p_0H(r_1) + p_1H(r_2) + p_2H(r_3) + p_3H(r_4), όπου με H(r_i) την εντροπία της i-οστής γραμμής, αριθμός υπολογίσιμος και ίδιος με την περίπτωση των ισοπίθανων συμβόλων εισόδου. Καλούμαστε να λύσουμε το πρόβλημα γραμμικού προγραμματισμού. max H(Y) – H(Y|X), όπου $p_0 + p_1 + p_2 + p_3$ = 1 και 0 ≤ p_0, p_1, p_2, p_3 ≤ 1. Με τη βοήθεια του μαθηματικού εργαλείου, το μέγιστο επιτυγχάνεται για την κατανομή (0.2574, 0.2488, 0.2129, 0.2809) και η ακριβής χωρητικότητα υπολογίζεται ως C = 1.0898 bits.

5.

Αν το κάθε σύμβολο μεταδιδόταν χωρίς λάθος, ο πίνακας μεταβάσεων θα ήταν ο παρακάτω:

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

Ο πίνακας είναι συμμετρικός και συνεπώς η χωρητικότητα υπολογίζεται ως: C = log_2|Y| - H(r) = $log_2 4$ = 2 bits.

Παράθεση Κώδικα σε Γλώσσα Προγραμματισμού Python

Άσκηση 1

Είσοδος:

1.

```
 from math import log
from scipy.optimize import minimize, Bounds, LinearConstraint
import matplotlib.pyplot as plt

def calculate_entropy(probs):
    return -sum(i*log(i,2) for i in probs if i !=0)

def objective_function(probs):
    p_0 = probs[0]
    p_1 = probs[1]
    p_2 = probs[2]
    H_r1 = calculate_entropy([1, 0, 0])
    H_r2 = calculate_entropy([0, 1, 0])
    H_r3 = calculate_entropy([(1/2)*(1-q),(1/2)*(1-q), q])
    H_y = 0
    if (1*p_0 + 0*p_1 + (1/2)*(1-q)*p_2 != 0):
      H_y -=(1*p_0 + 0*p_1 + (1/2)*(1-q)*p_2) * log((1*p_0 +
0*p_1 + (1/2)*(1-q)*p_2),2)

    if (0*p_0 + 1*p_1 + (1/2)*(1-q)*p_2 != 0):
      H_y -=(0*p_0 + 1*p_1 + (1/2)*(1-q)*p_2) * log((0*p_0 +
1*p_1 + (1/2)*(1-q)*p_2),2)
    if (0*p_0 + 0*p_1 + q*p_2 != 0):
      H_y -=(0*p_0 + 0*p_1 + q*p_2)* log((0*p_0 + 0*p_1 +
q*p_2),2)
    H_y_x = p_0* H_r1 + p_1 * H_r2 + p_2 * H_r3
    result = H_y - H_y_x
    return -result
 x0 = [1/3, 1/3, 1/3]

bounds = Bounds([0,0,0], [1,1,1])
```

```
linear_constraint = LinearConstraint([[1,1,1]], [1], [1])

for q in [0, 1/2, 1]:
  res = minimize(objective_function, x0,
constraints=[linear_constraint], bounds=bounds)
  print(f"Probability distribution is {res.x}")
  print(f"C = {-res.fun} for q =", q)
```

2.

```
Capacities=[]
Prob = []
q_values = [i/100 for i in range(101)]
for q in q_values:
    res = minimize(objective_function, x0,
constraints=[linear_constraint], bounds=bounds)
    Prob.append(res.x)
    Capacities.append(-res.fun)

plt.plot(q_values, Capacities)
plt.xlabel('q')
plt.ylabel('Capacity')
plt.title('Capacity vs q')
min_limit, max_limit = 0, 1
plt.xlim(min_limit, max_limit)
plt.show()

for i in range(len(Prob[0])):
    plt.plot(q_values, [prob[i] for prob in Prob],
label=f'Prob[{i}]')

plt.xlabel('q')
plt.ylabel('Probabilities')
plt.title('Probabilities vs q')
plt.legend()
min_limit, max_limit = 0, 1
plt.xlim(min_limit, max_limit)
plt.show()
```

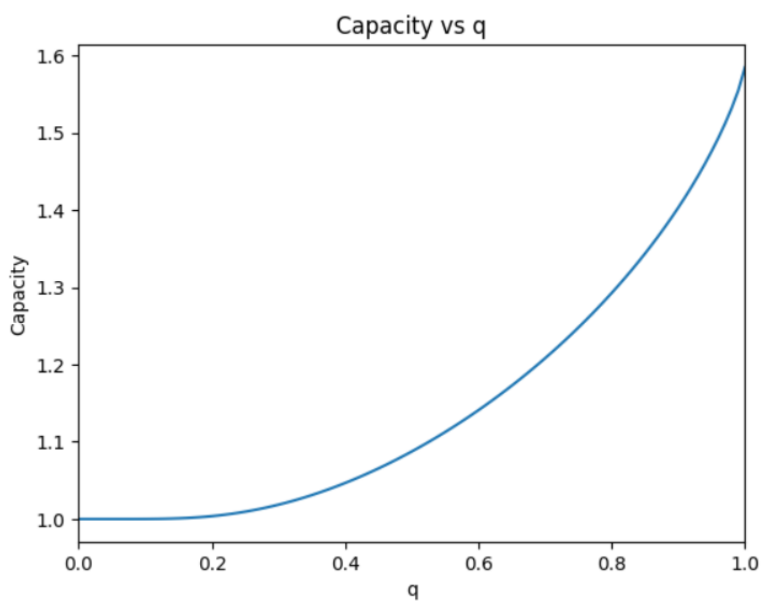
Έξοδος:

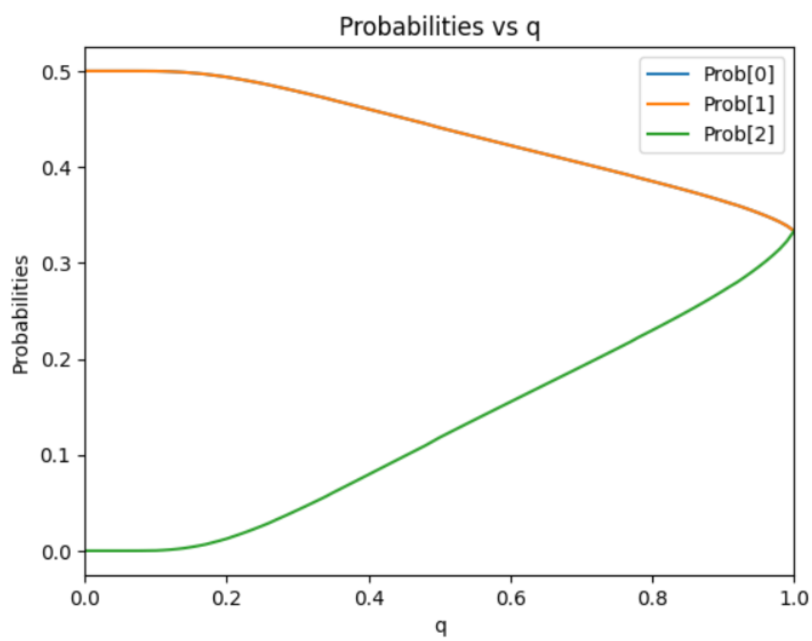
1.

```
Probability distribution is [0.5 0.5 0. ]

C = 1.0 for q = 0
Probability distribution is [0.44099396 0.44099398 0.11801206]
C = 1.0874624077049773 for q = 0.5
Probability distribution is [0.33333333 0.33333333 0.33333333]
C = 1.584962500721156 for q = 1
```

2.





<u>Άσκηση 2</u>

Είσοδος:

1.

```
from math import log
from scipy.optimize import minimize, Bounds, LinearConstraint

def contains_zero(lst):
    return 0 in lst

def calculate_entropy(probs):
    return -sum(i*log(i,2) for i in probs)

def objective_function(probs):
    p_0 = probs[0]
    p_1 = probs[1]
    p_2 = probs[2]
```

```
    p_3 = probs[3]
    H_r1 = calculate_entropy([0.8, 0.1, 0.1])
    H_r2 = calculate_entropy([0.05, 0.85, 0.05, 0.05])
    H_r3 = calculate_entropy([0.08, 0.06, 0.8, 0.06])
    H_r4 = calculate_entropy([0.05, 0.05, 0.05, 0.85])
    H_y = - (0.8*p_0 + 0.05*p_1 + 0.08*p_2 + 0.05*p_3) *
log((0.8*p_0 + 0.05*p_1 + 0.08*p_2 + 0.05*p_3),2) \
        - (0.1*p_0 + 0.85*p_1 + 0.06*p_2 + 0.05*p_3) *
log((0.1*p_0 + 0.85*p_1 + 0.06*p_2 + 0.05*p_3),2) \
        - (0.1*p_0 + 0.05*p_1 + 0.8*p_2 + 0.05*p_3)* log((0.1*p_0
+ 0.05*p_1 + 0.8*p_2 + 0.05*p_3),2) \
        - (0.0*p_0 + 0.05*p_1 + 0.06*p_2 + 0.85*p_3) *
log((0.0*p_0 + 0.05*p_1 + 0.06*p_2 + 0.85*p_3),2)
    H_y_x = p_0* H_r1 + p_1 * H_r2 + p_2 * H_r3 + p_3 * H_r4
    result = H_y - H_y_x
    return -result

probs_y = [0.245, 0.265, 0.250, 0.240]
transition_matrix = [
    [0.8, 0.1, 0.1, 0.0],
    [0.05, 0.85, 0.05, 0.05],
    [0.08, 0.06, 0.8, 0.06],
    [0.05, 0.05, 0.05, 0.85]
]


H_y = calculate_entropy(probs_y)
print("H(Y) =",H_y)
H_y_x = 0
for i in transition_matrix:
  if not contains_zero(i):
      H_y_x += (1/4) * calculate_entropy(i)
print("H(Y|X) =", H_y_x)
mutual_info = round(H_y - H_y_x, 5)
print(f"I(X;Y) = {mutual_info}")
```

2.

```
answer = round(2-calculate_entropy( [0.08, 0.06, 0.80, 0.06] ) ,
5 )

print (f"Channel Capacity: { answer }")
```

4.

```
x0 = [0.25, 0.25, 0.25, 0.25]

bounds = Bounds([0,0,0,0], [1,1,1,1])
linear_constraint = LinearConstraint([[1,1,1,1]], [1], [1])

res = minimize(objective_function, x0,
constraints=[linear_constraint], bounds=bounds)
print(f"Probability distribution is {res.x}")
print(f"C = {-res.fun}")
```

Έξοδος:

1.

```
H(Y) = 1.9989983098904922
H(Y|X) = 0.6828218933568609
I(X;Y) = 1.31618
```

2.

```
Channel Capacity: 0.96388
```

4.

```
Probability distribution is [0.25741439 0.24880208 0.2128828
0.28090073]
C = 1.089768438908867
```