Problem 1 (10 points)**.** Write a program that performs a Monte Carlo simulation of the transmission of equiprobable bits using (1, 1), (3, 1), (5, 1) and (7, 1) repetition codes over the BSC(p). Plot the simulated bit error probability as a function of $p \in [0, 1/2]$. Compute the the theoretical bit error probability in all four cases and plot in on the same plot.

Solution. For the $(2m+1, 1)$ repetition code, the theoretical bit error probability P_e is defined as $P_e \triangleq \mathbb{P}[\hat{X} \neq X]$, where X denotes the transmitted message bit and $\hat{X}$ denotes the decoded message bit. We assume that X is uniformly distributed over $\{0, 1\}$. Hence, we have $P_e = \mathbb{P}[\hat{X} = 1 | X = 0]$. Now, suppose $X = 0$ is the message to be sent, the repetition code encodes $X = 0$ as $2m + 1$ many 0's, which are sent over the BSC(p) channel. The majority decoding algorithm outputs $\hat{X} = 1$ if and only if more than m 0's are flipped to 1's, hence we have

$$P_e = \sum_{k=m+1}^{2m+1} \binom{2m+1}{k} p^k (1-p)^{2m+1-k}.$$

Figure 1 shows the simulation result. A sample code is also attached at the end. □

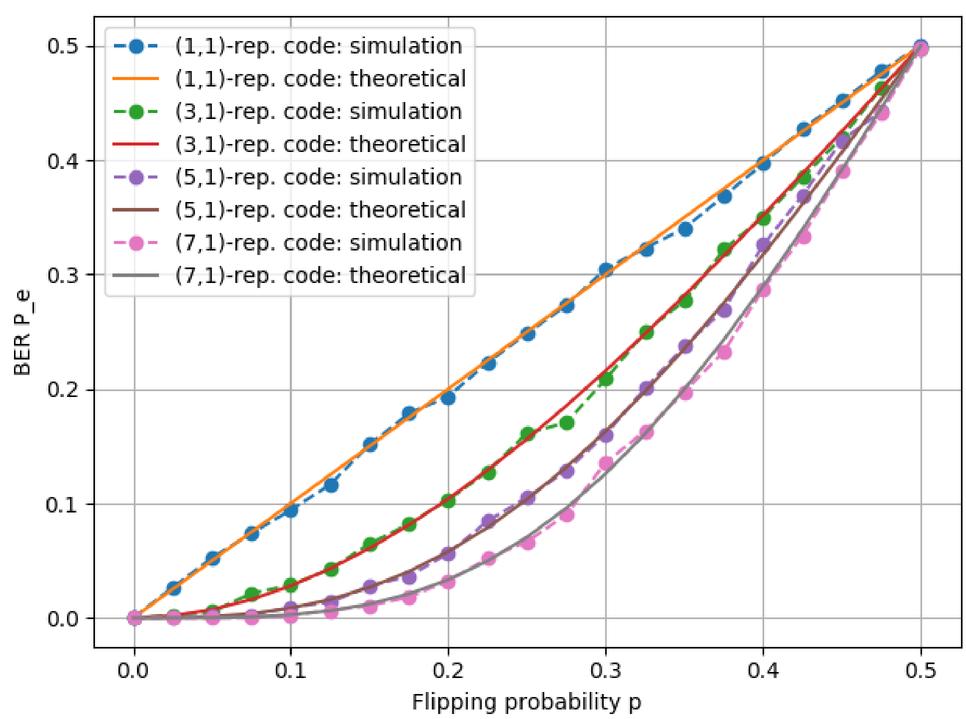

Figure 1: Simulation of repetition codes.

Problem 2 (10 points)**.** A card is selected equally likely from {K♠, 7♠, 7♡, K♣}. V and S denote the value and the suit of the selected card. Find the following entropies and conditional entropies of the value and the suit of the selected card (in bits)

$$H(V, S) \tag{1}$$
$$H(V) \tag{2}$$
$$H(S) \tag{3}$$
$$H(V|S = \clubsuit) \tag{4}$$
$$H(V|S = \spadesuit) \tag{5}$$
$$H(V|S) \tag{6}$$

Solution. The joint probability of (V, S) is represented by the following table:

	$S = \spadesuit$	$S = \heartsuit$	$S = \clubsuit$
$V = 7$	$\frac{1}{4}$	$\frac{1}{4}$	0
$V = K$	$\frac{1}{4}$	0	$\frac{1}{4}$

from which we can find all the marginal and conditional distributions. Hence, we have

$$\begin{aligned}
H(V, S) &= -4 \times \frac{1}{4} \log \frac{1}{4} = \log(4) = 2 \text{ bits} \\
H(V) &= -\frac{2}{4} \log \frac{2}{4} - \frac{2}{4} \log \frac{2}{4} = \log(2) = 1 \text{ bit} \\
H(S) &= -\frac{2}{4} \log \frac{2}{4} - \frac{1}{4} \log \frac{1}{4} - \frac{1}{4} \log \frac{1}{4} = \log(2\sqrt{2}) = 1.5 \text{ bits} \\
H(V|S = \clubsuit) &= H((0, 1)) = 0 \text{ bits} \\
H(V|S = \spadesuit) &= H((1/2, 1/2)) = \log(2) = 1 \text{ bit} \\
H(V|S) &= \frac{1}{2} H(V|S = \spadesuit) + \frac{1}{4} H(V|S = \heartsuit) + \frac{1}{4} H(V|S = \clubsuit) \\
&= \frac{1}{2} \log(2) + \frac{1}{4} \times 0 + \frac{1}{4} \times 0 \\
&= \log(\sqrt{2}) = 0.5 \text{ bits}
\end{aligned}$$

□

Problem 3 (10 points)**.** For a discrete random variable X, the Rényi entropy of order $\alpha > 0, \alpha \neq 1$ is defined as

$$H_\alpha(X) = \frac{1}{1-\alpha} \log \left(\sum_{a \in \mathcal{A}} P_X^\alpha(a) \right) \tag{7}$$

1. Find the maximum value that $H_\alpha(X)$ can take over all random variables X taking M different values.

2. If X and Y are independent, show that

$$H_\alpha(X, Y) = H_\alpha(X) + H_\alpha(Y) \tag{8}$$

Solution.

1. For each $i = 1, ..., M$, let p_i denote the probability of $X = i$. Then, we rewrite $H_\alpha(X)$ as a function of the probability mass function $p \in \mathcal{S}$, where

$$\mathcal{S} \triangleq \left\{ p \in \mathbb{R}^M : p_i \geq 0, \quad \sum_{i=1}^{M} p_i = 1 \right\}.$$

We denote the Rényi entropy as a function $H_\alpha(p) : \mathcal{S} \mapsto \mathbb{R}$.

(a) (Method 1: $H_\alpha(p)$ is non-increasing as α increases) Observe that for any *fixed* distribution p, the Rényi entropy $H_\alpha(p)$ is a non-increasing function of $\alpha \geq 0$. The reason is that

$$\begin{aligned}
\frac{\partial H_\alpha(p)}{\partial \alpha} &= \frac{1}{(1-\alpha)^2} \left((1-\alpha) \frac{\sum_i p_i^\alpha \log(p_i)}{\sum_j p_j^\alpha} - (-1) \log \sum_j p_j^\alpha \right) \\
&= -\frac{1}{(1-\alpha)^2} \left(-(1-\alpha) \sum_i z_i \log(p_i) - \sum_i z_i \log \sum_j p_j^\alpha \right) \\
&= -\frac{1}{(1-\alpha)^2} \left(\sum_i z_i \log \frac{p_i^{\alpha-1}}{\sum_j p_j^\alpha} \right) \\
&= -\frac{1}{(1-\alpha)^2} \left(\sum_i z_i \log \frac{z_i}{p_i} \right) \\
&= -\frac{1}{(1-\alpha)^2} D(z||p) \\
&\leq 0,
\end{aligned}$$

where

$$z_i \triangleq \frac{p_i^\alpha}{\sum_j p_j^\alpha},$$

and $D(z||p)$ is the Kullback–Leibler divergence from z to p. The nonnegativity of

$D(z||p)$ can be proved by using Jensen's inequality:

$$\begin{aligned} D(z||p) &\triangleq \sum_i z_i \log \frac{z_i}{p_i} \\ &= -\mathbb{E}_{X\sim z} \log \frac{p_X}{z_X} \\ &\geq -\log \mathbb{E}_{X\sim z} \frac{p_X}{z_X} \\ &= -\log \sum_i p_i \\ &= 0, \end{aligned}$$

where X is a random variable taking values in $\{1, ..., M\}$ with distribution given by z.

Hence, for any *fixed* distribution p, we have $H_\alpha(p) \leq H_0(p) = \log M$. On the other hand, for any *fixed* $\alpha > 0$ and $\alpha \neq 1$, the Rényi entropy of the uniform distribution is

$$H_\alpha\left(\frac{1}{M}\mathbf{1}\right) = \frac{1}{1-\alpha} \log \sum_{i=1}^{M} \frac{1}{M^\alpha} = \log M,$$

which achieves the upper bound $\log M$. Hence, for any $\alpha > 0$ and $\alpha \neq 1$, the the uniform distribution maximizes the Rényi entropy.

(b) (Method 2: Concavity + permutation invariance) First, we show that $H_\alpha(p)$ is a concave function in p for any $\alpha > 0$ and $\alpha \neq 1$. We observe that the function $x \mapsto x^\alpha$ is concave in x when $\alpha \in (0, 1)$; and is convex when $\alpha > 1$. In addition, the function $y \mapsto \log y$ is monotonically increasing and thus preserves convexity and concavity. Therefore, $H_\alpha(p)$ is concave in p in both cases.

Next, we observe that $H_\alpha(p)$ is permutation invariant in p: for any permutation matrix Π, we have $H_\alpha(\Pi \times p) = H_\alpha(p)$ for any $p \in \mathcal{S}$, where $\Pi \times p$ is a vector obtained from p by permuting the coordinates of p by Π. Clearly, we have $\Pi \times p \in \mathcal{S}$.

For any $p \in \mathcal{S}$, let Π_k denote all $M!$ permutation matrices (or, the M circular-shift permutation matrices would also work), then we have

$$\frac{1}{M!} \sum_{k=1}^{M!} (\Pi_k \times p) = \frac{1}{M}\mathbf{1},$$

where $\mathbf{1}$ denotes the all-one vector. By concavity of $H_\alpha(p)$, we have

$$\begin{aligned}
H_\alpha\left(\frac{1}{M}\mathbf{1}\right) &= H_\alpha\left(\frac{1}{M!}\sum_{k=1}^{M!}\Pi_k \times p\right) \\
&\geq \frac{1}{M!}\sum_{k=1}^{M!} H_\alpha\left(\Pi_k \times p\right) \\
&= \frac{1}{M!}\sum_{k=1}^{M!} H_\alpha\left(p\right) \\
&= H_\alpha\left(p\right),
\end{aligned}$$

where the first inequality is by Jensen's inequality; the second equality is by permutation invariance. Hence, the uniform distribution $\frac{1}{M}\mathbf{1}$ maximizes $H_\alpha(p)$:

$$H_\alpha\left(\frac{1}{M}\mathbf{1}\right) = \log(M).$$

(c) (Method 3: Checking KKT conditions) We are solving the following maximization problem:

$$H^\star \triangleq \max_{p\in\mathcal{S}} \frac{1}{1-\alpha}\log\left(\sum_{i=1}^{M} p_i^\alpha\right).$$

For $\alpha \in (0,1)$, since $1-\alpha > 0$ and log is monotonically increasing, finding $H^\star$ is equivalent to solve the maximization of a concave function in p:

$$f^\star \triangleq \max_{p\in\mathcal{S}} \sum_{i=1}^{M} p_i^\alpha.$$

For $\alpha > 1$, finding $H^\star$ is equivalent to solve the minimization of a convex function in p:

$$g^\star \triangleq \min_{p\in\mathcal{S}} \sum_{i=1}^{M} p_i^\alpha.$$

In both cases, we have a standard convex optimization problem (that is, minimization of a convex function, or maximization of a concave function, over convex sets), the difficulty is that these two in current forms are constrained optimization problem. Thus we appeal to Lagrangian duality, and form the Lagrangian

$$L(p,\lambda,\mu) \triangleq \sum_{i=1}^{M} p_i^\alpha - \lambda' p + \mu(\mathbf{1}'p - 1),$$

where $\lambda \in \mathbb{R}^M$ and $\lambda_i \geq 0$, $\mu \in \mathbb{R}$, and $\mathbf{1}$ denotes the all-one vector. Since the Slater's condition holds (that is, the constrained set $\mathcal{S}$ has nonempty relative interior, e.g. the uniform distribution $\mathbf{1}/M$ is in the relative interior of $\mathcal{S}$), we have strong duality, that is,

$$\begin{aligned} f^\star &= \max_{p \in \mathbb{R}^M} \min_{\lambda \geq 0, \mu \in \mathbb{R}} L(p, \lambda, \mu) \\ &= \min_{\lambda \geq 0, \mu \in \mathbb{R}} \max_{p \in \mathbb{R}^M} L(p, \lambda, \mu), \end{aligned}$$

where the last equality is from the strong duality. Note that by doing so, we notice that the inner maximization becomes unconstrained and we can take derivatives. Define

$$h(\lambda, \mu) \triangleq \max_{p \in \mathbb{R}^M} L(p, \lambda, \mu),$$

and take derivatives: for $j = 1, ..., M$,

$$\frac{\partial L(p, \lambda, \mu)}{\partial p_j} = \alpha p_j^{\alpha-1} - \lambda_j + \mu.$$

Let $p^\star$ be the primal optimizer and $(\lambda^\star, \mu^\star)$ be the dual optimizer. Then, from the Karush-Kuhn-Tucker conditions (KKT conditions, necessary conditions for optimality), we know that $p^\star, \lambda^\star, \mu^\star$ must satisfy:

$$\begin{aligned} \alpha (p_j^\star)^{\alpha-1} - \lambda_j^\star + \mu^\star &= 0, \ \forall j \\ \lambda_j^\star p_j^\star &= 0, \ \forall j \\ \lambda_j^\star &\geq 0, \ \forall j \\ \mathbf{1}' p^\star &= 1 \\ p_j^\star &\geq 0, \ \forall j \end{aligned}$$

By the first and last conditions, we must have $\lambda_j^\star - \mu^\star \geq 0$. There are two cases. Case (1): There exists a j such that $\lambda_j^\star - \mu^\star > 0$, then we have

$$p_j^\star = \left(\frac{\lambda_j^\star - \mu^\star}{\alpha} \right)^{\frac{1}{\alpha-1}} > 0,$$

which combined with the second condition implies that $\lambda_j^\star = 0$ and $\mu^\star < 0$ for such a j. Hence, we can further simplify $p_j^\star$ as

$$p_j^\star = \left(\frac{-\mu^\star}{\alpha} \right)^{\frac{1}{\alpha-1}}.$$

Case (2): There exists a j such that $\lambda_j^\star - \mu^\star = 0$, then by the first condition we have $p_j^\star = 0$ and $\mu^\star \geq 0$. Notice that these two cases are mutually exclusive since $\mu^\star < 0$ in the first case while $\mu^\star \geq 0$ in the second case. That means, either $\lambda_j^\star = 0$ for all j's, or $p_j^\star = 0$ for all j's. By the fourth condition, the later case is not possible. Therefore, we must have for any $j = 1, ..., M$, $\lambda_j^\star = 0$ and $\mu^\star < 0$, and all $p_j^\star$'s are the same:

$$\begin{aligned}
p_j^\star &= \frac{1}{M} \\
\mu^\star &= -\alpha M^{1-\alpha}.
\end{aligned}$$

Therefore, the uniform distribution maximizes $H_\alpha(X)$ for $\alpha \in (0,1)$. The same argument yields that the uniform distribution also maximizes $H_\alpha(X)$ for $\alpha > 1$.

2. For independent X and Y, we have

$$\begin{aligned}
H_\alpha(X,Y) &= \frac{1}{1-\alpha} \log \sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} P_{X,Y}^\alpha(x,y) \\
&= \frac{1}{1-\alpha} \log \sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}} P_X^\alpha(x) P_Y^\alpha(y) \\
&= \frac{1}{1-\alpha} \log \left(\sum_{x\in\mathcal{X}} P_X^\alpha(x) \sum_{y\in\mathcal{Y}} P_Y^\alpha(y) \right) \\
&= \frac{1}{1-\alpha} \log \left(\sum_{x\in\mathcal{X}} P_X^\alpha(x) \right) + \frac{1}{1-\alpha} \log \left(\sum_{y\in\mathcal{Y}} P_Y^\alpha(y) \right) \\
&= H_\alpha(X) + H_\alpha(Y).
\end{aligned}$$

□

```
import numpy as np
from scipy.stats import bernoulli
from scipy.stats import binom
import matplotlib.pyplot as plt

# EE160 HW1 Problem 1
# By Peida Tian
def encoder(b, n):
    return [b for i in range(n)]

def decoder(y):
    d = {0:0, 1:0}
    for b in y:
        d[b] += 1
    x = 0
    if d[1] > d[0]:
        x = 1
    return x

def bsc(p, x):
    y = []
    n = len(x)
    err = bernoulli.rvs(size=n, p=p)
    y = list((np.array(x) + err) % 2)
    return y

M = [0, 1, 2, 3]
ML = len(M)
# N: number of simulations,
# that is, number of message bits to be sent
N = 5000

L = 20
P = [0.5*i / L for i in range(L+1)]
K = len(P)
res = []

# Simulation
for j in range(ML):
    m = M[j]
    n = 2*m+1
    res.append([])
    for i in range(K):
        p = P[i]
        # generate messages
        s = bernoulli.rvs(size=N, p=0.5)
        s_hat = []
        s_list = list(s)
        errRate = 0.0
        for message in s_list:
            x = encoder(message, n)
            y = bsc(p, x)
            message_hat = decoder(y)
```

```
            errRate += abs(message - message_hat)

        errRate /= N
        res[j].append(errRate)

# Theoretical Probability of bit error
Pe = []
for j in range(ML):
    m = M[j]
    n = 2*m+1
    Pe.append([])
    for p in P:
        Pe[j].append(1.0 - binom.cdf(m, n, p, loc=0))

# Plot
legend_labels = []
for j in range(ML):
    m = M[j]
    n = 2*m+1
    label_sim = "(" + str(n) + "," + "1)-rep. code: simulation"
    label_th = "(" + str(n) + "," + "1)-rep. code: theoretical"
    simulation, = plt.plot(P, res[j], label=label_sim, linestyle='--', ...
        marker='o')
    theoretical, = plt.plot(P, Pe[j], label=label_th)
    legend_labels.append(simulation)
    legend_labels.append(theoretical)

plt.legend(handles=legend_labels)
plt.xlabel('Flipping probability p')
plt.ylabel('BER P_e')
plt.grid()
plt.show()
```

Note: For the programming exercises, you may use any programming language. You may not use any of the built-in library functions that perform coding for you, such as Matlab's `encode` and `decode` functions. Please supply your code with enough comments for your TA to understand.