



به نام خدا

دانشگاه صنعتی شریف

دانشکده‌ی مهندسی برق

سیستم‌های مخابراتی - گروه دکتر پاک‌روان

نیم‌سال اول ۱۴۰۱-۱۴۰۲

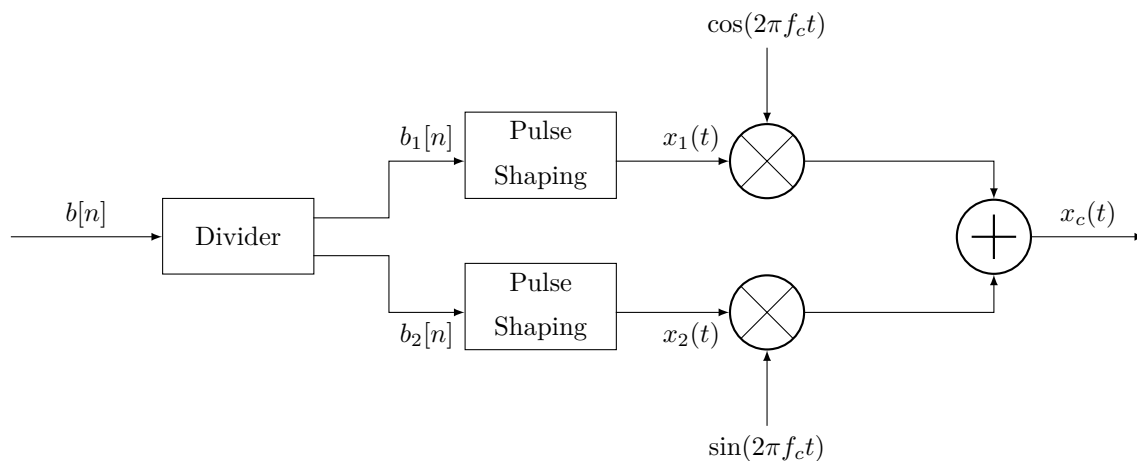
پروژه‌ی درس سیستم‌های مخابراتی

لطفاً به نکات زیر توجه کنید:

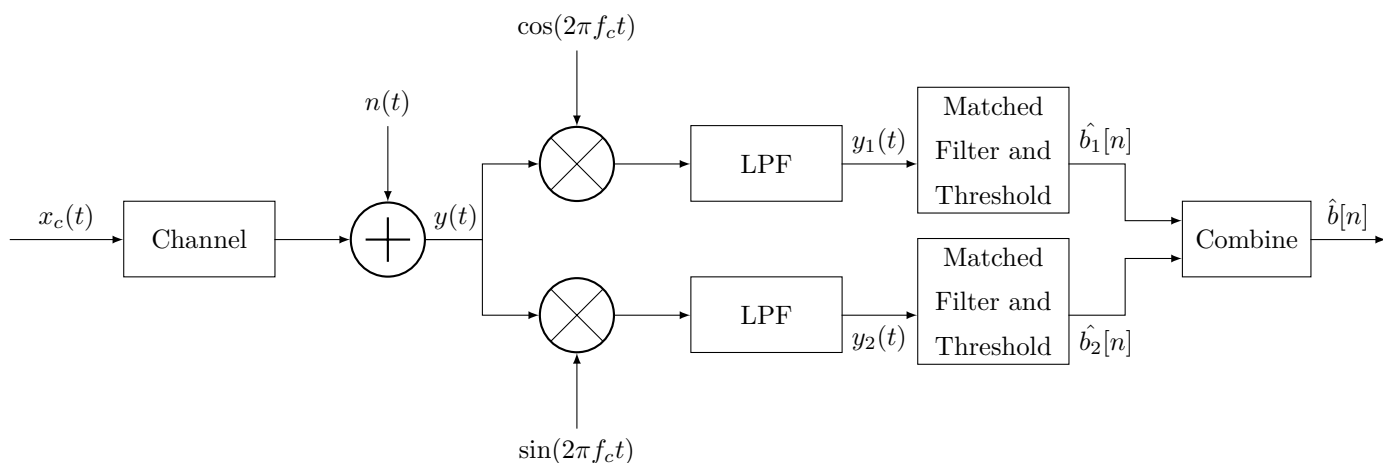
۱. این پروژه امتیازی و اختیاری است.
۲. شبیه‌سازی‌ها را می‌توانید با کمک MATLAB یا Python انجام دهید.
۳. نتایج و پاسخ‌های خود را در یک فایل با فرمت zip به نام Project_Name_StudentNumber در سایت CW قرار دهید.
۴. تحویل پروژه به صورت گزارش و کدهای نوشته‌شده است. در گزارش باید روشی که استفاده کرده‌اید را توضیح و نتایجی که گرفته‌اید را ارائه دهید. همچنین گزارش باید شامل تصاویر و نمودارها و نتیجه‌گیری‌های لازم باشد. همچنین تمیزی گزارش بسیار مهم است. کدها و گزارش را در یک فایل فشرده‌شده در سامانه‌ی درس‌افزار آپلود کنید.
۵. کدهای خود را خوانا بنویسید و کامنت‌گذاری کنید.
۶. ابهام یا اشکالات خود را می‌توانید از طریق @Hamedajorlou یا Hamedajorloo2@gmail.com مطرح نمایید.
۷. کدهای شما تماماً باید توسط خودتان نوشته شده باشند. هرگونه استفاده از کد دیگران به هر شکل ممکن، تقلب محسوب می‌شود و نمره‌ی پروژه صفر خواهد شد. پس در هیچ صورت کدهای خود را برای دیگران ارسال نکنید.
۸. مهلت تحویل: ۱۴ بهمن ۱۴۰۱

۱ مقدمه

در این پروژه قصد داریم یک سیستم مخابرات دیجیتال را به طور کامل شبیه‌سازی کنیم و تأثیر پارامترهای مختلف را بر عملکرد این سیستم مشاهده کنیم. دیاگرام بلوکی فرستنده و گیرنده در شکل‌های ۱ و ۲ نمایش داده شده‌اند. برای راحتی، در این پروژه فقط حالت باینری را در نظر می‌گیریم.



شکل ۱: دیاگرام بلوکی فرستنده



شکل ۲: دیاگرام بلوکی گیرنده

۲ پیاده‌سازی بلوک‌ها به صورت مجزاً

۱. تابعی با عنوان Divide بنویسید، که در ورودی دنباله‌ای به طول زوج از اعداد صفر و یک بگیرد، و در خروجی دو دنباله به طول نصف دنباله‌ی ورودی تحویل دهد. (به نظرتان باید این تابع چگونه دنباله‌ی ورودی را تقسیم کند که شبیه‌سازی ما به یک سیستم real-time نزدیک‌تر شود؟) وارون این سیستم را در تابعی به نام Combine پیاده‌سازی کنید.

۲. تابعی با عنوان PulseShaping بنویسید که در ورودی، دنباله‌ای از صفر و یک، شکل پالس متناظر با صفر (به صورت رشته‌ای از اعداد حقیقی!)، و شکل پالس متناظر با یک (به صورت رشته‌ی دیگری از اعداد حقیقی!) را دریافت کند، و در خروجی، شکل موج متناظر با دنباله را تحویل دهد. توجه کنید که طول پالس‌های متناظر با صفر و یک باید برابر باشند.

۳. تابعی با عنوان AnalogMod بنویسید که در ورودی، دو شکل موج، فرکانس نمونه‌برداری و فرکانس حامل را دریافت کرده و در خروجی سیگنال $x_c(t)$ را بدهد.

۴. برای سادگی، کانال را ایده‌آل در نظر می‌گیریم و فقط پهنای باند سیگنال عبوری از آن را محدود می‌کنیم. برای حل این بخش، تابعی با عنوان Channel بنویسید که در ورودی سیگنال ارسالی، فرکانس نمونه‌برداری، فرکانس مرکزی و پهنای باند کانال را بگیرد و در خروجی، سیگنال دریافتی در گیرنده را تحویل دهد. (تنها باید یک فیلتر میان‌گذر بنویسید!)

۵. تابعی با عنوان AnalogDemod بنویسید که در ورودی سیگنال $x_c(t)$ ، فرکانس نمونه‌برداری، پهنای باند سیگنال و فرکانس حامل را بگیرد و در خروجی دو شکل موج demodulate شده (یعنی $y_1(t)$ و $y_2(t)$ در شکل ۲) را تحویل دهد. (ترکیب ضرب‌کننده و فیلتر پایین‌گذر)

۶. برای آخرین قسمت نیز، تابعی با عنوان MatchedFilt بنویسید که در ورودی شکل موج demodulate شده، شکل پالس متناظر با صفر، و شکل پالس متناظر با یک را دریافت کند و در خروجی دو دنباله به این صورت بدهد:

(الف) مقدار خروجی Matched Filter هم برای شکل پالس متناظر با یک و هم برای شکل پالس متناظر با صفر (در اسلایدهای درس با Matched Filter آشنا شده‌اید و می‌دانید که برای یک شکل پالس دلخواه، Matched Filter به چه صورت درمی‌آید!)

(ب) مقدار تخمین زده‌شده برای بیت متناظر

۳ انتقال دنباله‌ی تصادفی صفر و یک

برای این قسمت، مشخصات بلوک‌ها را به این صورت در نظر بگیرید:

مقدار	متغیر
1 MHz	فرکانس نمونه‌برداری
10 ms	طول هر پالس
10 KHz	فرکانس حامل
10 KHz	فرکانس مرکزی کانال
1 KHz	پهنای باند کانال

جدول ۱: مشخصات بلوک‌ها

۱. در این قسمت می‌خواهیم با استفاده از مدولاسیون PAM سیگنال‌ها را ارسال کنیم. شکل پالس را مربعی ساده در نظر بگیرید که دامنه‌اش برای ارسال بیت یک، برابر با +1 و برای بیت صفر، برابر با -1 است. (هر پارامتر دیگری که قرار است انتخاب کنید را به شکلی معقول انتخاب کنید، و دلایلتان را برای آن انتخاب شرح دهید.)

(الف) دنباله‌ای به طول «به اندازه‌ی کافی بلند» از صفر و یک تولید کنید. با فرض وجود نداشتن نویز، فرآیند ارسال این دنباله را شبیه‌سازی کنید. شکل موج را برای خروجی هر بلوک رسم کنید. (بازه‌ی زمانی رسم‌کردن را به شکل معقولی در نظر بگیرید که شکل‌های شما «قشنگ» باشند!)

(ب) در درس کمی با فرآیندهای تصادفی آشنا شده‌اید. یکی از مهم‌ترین فرآیندهای تصادفی، نویز است. ما در اغلب اوقات فرض می‌کنیم که نویز با سیگنال جمع می‌شود و به تعبیر دیگر، افزایشده (Additive) است. همچنین لحظات مختلف نویز را از یک دیگر مستقل فرض می‌کنیم، با به تعبیر دیگر، نویز را سفید (White) فرض می‌کنیم. به عنوان فرض آخر، توزیع نویز در همه‌ی لحظات را گوسی با میانگین صفر فرض می‌کنیم. چنین نویزی را Additive White Gaussian Noise می‌نامیم.

حال با فرض اینکه نویز AWGN، بعد از عبور سیگنال از کانال با آن جمع می‌شود، احتمال خطا را برحسب واریانس نویز رسم کنید. بازه‌ی محور افقی را به اندازه‌ی کافی بزرگ بگیرید. رفتار احتمال خطا را توجیه کنید.

(ج) با توجه به بخش قبل، ۶ مقدار مختلف برای واریانس نویز انتخاب کنید، (به صورتی که تا حد خوبی بازه‌های معنادار نمودار را پوشش دهد). برای هر کدام از این واریانس‌ها، scatter plot دوبعدی خروجی دو Matched Filter را رسم کنید. این به آن معناست که به ازای هر زوج $(b_1[i], b_2[i])$ ، به علت وجود نویز، خروجی دو Matched Filter به صورت دو عدد $(\hat{b}_1[i], \hat{b}_2[i])$ درمی‌آیند که احتمالاً $\hat{b}_1[i] \neq b_1[i]$ ، $\hat{b}_2[i] \neq b_2[i]$ ولی اگر مقدار واریانس نویز کم باشد، می‌توان از روی $(\hat{b}_1[i], \hat{b}_2[i])$ ، $(b_1[i], b_2[i])$ را به درستی تخمین زد. خواسته‌ی مسأله آنست که $(\hat{b}_1[i], \hat{b}_2[i])$ را به صورت یک نقطه در فضای دوبعدی تصور کنید و به ازای i ‌های مختلف، همه‌ی این نقاط را روی یک نمودار رسم کنید. این کار را برای ۶ مقدار واریانس نویزی که انتخاب کرده‌اید، تکرار کنید. برای آشنایی بهتر با خواسته‌ی مسأله، می‌توانید عبارت «منظومه‌ی سیگنال» (Signal Constellation) را جستجو کنید.

۲. در این بخش به سراغ مدولاسیون PSK می‌رویم. به جای شکل پالس مربعی، شکل پالس سینوسی با فرکانس 500 Hz در نظر بگیرید. دامنه‌ی آن را برای ارسال بیت یک، برابر با +1 و برای ارسال بیت صفر، برابر با -1 فرض کنید.

(الف) دنباله‌ای به طول «به اندازه‌ی کافی بلند» از صفر و یک تولید کنید. با فرض وجود نداشتن نویز، فرآیند ارسال این دنباله را شبیه‌سازی کنید. شکل موج را برای خروجی هر بلوک رسم کنید. (بازه‌ی زمانی رسم‌کردن را به شکل معقولی در نظر بگیرید که شکل‌های شما «قشنگ» باشند!)

(ب) با فرض اینکه نویز AWGN، بعد از عبور سیگنال از کانال با آن جمع می‌شود، احتمال خطا را برحسب واریانس نویز رسم کنید. بازه‌ی محور افقی را به اندازه‌ی کافی بزرگ بگیرید. رفتار احتمال خطا را توجیه کنید.

(ج) با توجه به بخش قبل، ۶ مقدار مختلف برای واریانس نویز انتخاب کنید، (به صورتی که تا حد خوبی بازه‌های معنادار نمودار را پوشش دهد). برای هر کدام از این واریانس‌ها، scatter plot دوبعدی خروجی دو Matched Filter را رسم کنید.

۳. در این بخش می‌خواهیم از مدولاسیون FSK استفاده کنیم. در نتیجه از پالس سینوسی با فرکانس متناسب با مقدار بیت استفاده می‌کنیم. فرکانس آن را برای ارسال بیت ۱، برابر 1 KHz و برای ارسال بیت صفر، برابر با 1.5 KHz فرض کنید.

(الف) آیا فرکانس‌های داده‌شده، یک سیگنالینگ متعامد را می‌سازند؟

(ب) دنباله‌ای به طول «به اندازه‌ی کافی بلند» از صفر و یک تولید کنید. با فرض وجود نداشتن نویز، فرآیند ارسال این دنباله را شبیه‌سازی کنید. شکل موج را برای خروجی هر بلوک رسم کنید. (بازه‌ی زمانی رسم‌کردن را به شکل معقولی در نظر بگیرید که شکل‌های شما «قشنگ» باشند!)

(ج) با فرض اینکه نویز AWGN، بعد از عبور سیگنال از کانال با آن جمع می‌شود، احتمال خطا را برحسب واریانس نویز رسم کنید. بازه‌ی محور افقی را به اندازه‌ی کافی بزرگ بگیرید. رفتار احتمال خطا را توجیه کنید.

(د) با توجه به بخش قبل، ۶ مقدار مختلف برای واریانس نویز انتخاب کنید، (به صورتی که تا حد خوبی بازه‌های معنادار نمودار را پوشش دهد). برای هر کدام از این واریانس‌ها، scatter plot دوبعدی خروجی دو Matched Filter را رسم کنید.

۴. نتایج این سه سیستم را باهم مقایسه کنید.

۴ انتقال دنباله‌ای از اعداد ۸ بیتی

در این قسمت می‌خواهیم دنباله‌ای از اعداد تصادفی بین ۰ و ۲۵۵ را به دنباله‌ای از اعداد ۰ و ۱ تبدیل کرده و انتقال آن‌ها را شبیه‌سازی کنیم. تفاوت اصلی این بخش با بخش قبلی در معیار سنجش صحت سیستم مخابره است. در این قسمت از مجذور اختلاف اعداد به جای احتمال خطا استفاده خواهیم کرد.

۱. تابعی با عنوان SourceGenerator بنویسید که دنباله‌ای از اعداد صحیح بین ۰ تا ۲۵۵ را بگیرد و در خروجی، دنباله‌ی باینری متناظر با آن را بدهد. (می‌توانید از تابع de2bi در MATLAB استفاده کنید). سیستم معکوس این تابع را نیز به صورت تابعی با عنوان OutputDecoder بنویسید. (می‌توانید از تابع bi2de در MATLAB استفاده کنید).

۲. دنباله‌ای به حد کافی بلند از اعداد صحیح بین ۰ تا ۲۵۵ تولید کنید. مخابره‌ی آن‌ها را شبیه‌سازی کنید. (شکل پالس‌ها را مرتب‌ی بگیرید و مشخصات سیستم را مطابق قسمت قبل) واریانس خطای بازسازی این اعداد را برحسب واریانس نویز رسم کنید.

۳. در ادامه‌ی بخش قبل، به ازای چند واریانس نویز مشخص، توزیع خطا را رسم کنید. رفتار حدی این توزیع به چه شکلی است؟

۴. در حالتی که نویز به سمت بی‌نهایت میل کند، واریانس خطا را به روش تحلیلی حساب کنید. آیا با آنچه که مشاهده می‌کنید سازگار است؟

۵ کدینگ منبع

در این قسمت می‌خواهیم با کدینگ منبع (Source Coding) آشنا شویم. خبر خوب آنست که این بخش از بخش‌های قبلی پروژه مستقل است!

ابتدا باید با مفاهیم اولیه آشنا شویم. منبع اطلاعات گسسته‌ی X که دنباله‌ی سمبل‌های X_i را تولید می‌کند در نظر می‌گیریم. فرض می‌کنیم که الفبای خروجی این منبع، \mathcal{X} است و فرض می‌کنیم $|\mathcal{X}| = M < \infty$.

برای مدل کردن منبع اطلاعات، باید به هر سمبل از منبع یک احتمال نسبت بدهیم. در نتیجه می‌توانیم منبع اطلاعات را به این صورت نمایش دهیم:

$$X = \begin{pmatrix} x_1 & x_2 & \dots & x_M \\ p_1 & p_2 & \dots & p_M \end{pmatrix}$$

به عنوان مثال اگر شما که به زبان فارسی حرف می‌زنید را به عنوان منبع اطلاعات در نظر بگیریم، هر سمبل خروجی این منبع، حرفی از حروف فارسی است. در نتیجه $\mathcal{X} = \{\text{ی}, \dots, \text{ب}, \text{الف}\}$ و $|\mathcal{X}| = M = 32$. همچنین به طور تجربی می‌دانیم که برخی حروف در زبان فارسی بیشتر استفاده می‌شوند، مانند «الف» و «م»، و برخی حروف کمتر استفاده می‌شوند مانند «ژ» و «ض». در نتیجه در مدل احتمالاتی این منبع، باید به حروف پر استفاده احتمال بیشتری نسبت به حروف کم استفاده نسبت دهیم.

(البته ذکر این نکته لازم است که شما احتمالاً کاملاً تصادفی صحبت نمی‌کنید و میان حروف استفاده شده توسط شما ارتباطاتی وجود دارد! به تعبیر دیگر حرف تولید شده در زمان $t-1$ اطلاعاتی درباره‌ی حرف تولید شده در زمان t در اختیار ما قرار می‌دهد و می‌توانیم $\mathbb{P}[X_t = x_i | X_{t-1} = x_j]$ را تشکیل دهیم و عملاً مدل دقیق‌تری برای منبع اطلاعات به دست آوریم که به صورت یک «زنجره‌ی مارکف» است. ولی در مدل فعلی فرض می‌کنیم که همه‌ی حروف تولیدی توسط شما از یکدیگر مستقلند و توزیع تمامی آن‌ها باهم برابر است.)

مسئله‌ی کدینگ منبع آنست که می‌خواهیم یک نگاشت مانند C از الفبای خروجی منبع به الفبای دوتایی $D = \{0, 1\}$ پیدا کنیم. (البته در حالت کلی‌تر الفبای کد می‌تواند D سمبلی باشد، ولی ما حالت ساده‌ی دوتایی را در نظر می‌گیریم.) به گونه‌ای که طول متوسط هر کلمه‌کد کمینه شود. به تعبیر دیگر اگر طول کلمه‌کد متناظر با سمبل $x \in \mathcal{X}$ را با $l(x)$ نشان دهیم، هدف آنست که $\mathbb{E}[l(X)]$ کمینه شود.

طبعاً ساده‌ترین کار ممکن آنست که به هر کدام از M خروجی منبع، یک عدد باینری $\lceil \log_2 M \rceil$ بیتی نسبت بدهیم، ولی این راه حل اصلاً بهینه نیست. چرا؟

مثال سخنران را در نظر بگیریم. در این مثال چون $M = 32$ است، با ۵ بیت می‌توان همه‌ی حروف را کد کرد. ولی در این حالت، ما برای حرف «الف» که بسیار پر استفاده است، ۵ بیت خرج کرده‌ایم و برای حرف «ژ» هم ۵ بیت! اگر راهی باشد که بتوانیم حرف «الف» را با کمتر از ۵ بیت کد کنیم و حرف «ژ» را با کمی بیشتر از ۵ بیت، از آنجا که احتمال بیان شدن حرف «الف» توسط سخنران بسیار بیشتر از احتمال بیان شدن حرف «ژ» است، انتظار داریم که به طور متوسط کمتر از ۵ بیت برای کد کردن منبع خرج کنیم و به کد بهینه‌تری برسیم.

احتمالاً اولین سؤالی که به ذهن می‌رسد، آنست که «اگر طول کلمه‌کدها برابر نباشد، در گیرنده چگونه آن‌ها را کدگشایی کنیم؟» داستان از همین جا شروع می‌شود!

منبع اطلاعات زیر را در نظر بگیرید:

$$X = \begin{pmatrix} a & b & c & d & e & f \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{8} & \frac{1}{16} & \frac{1}{32} & \frac{1}{32} \end{pmatrix}$$

۱. کدهای زیر را در نظر بگیرید:

X	a	b	c	d	e	f
$C_1(X)$	0	10	11	10	01	111
$C_2(X)$	0	10	01	010	11	111
$C_3(X)$	0	01	011	0111	01111	011111

(الف) مشکل کد C_1 چیست؟

کدهایی که مشکل کد C_1 را نداشته باشند، کدهای «ناویژه» می‌نامیم.

(ب) مشکل کد C_2 چیست؟ (راهنمایی: فرض کنید در گیرنده دنباله‌ی 010 دریافت شده. دنباله‌ی سمبل‌های ارسالی چگونه بوده است؟)

کدهایی که مشکل کد C_2 را نداشته باشند، کدهای «به طور یکتا قابل کشف» می‌نامیم.

(ج) مشکل کد C_3 چیست؟ (راهنمایی: آیا با مشاهده‌ی کلمه‌ی کد مربوط به سمبل a در گیرنده، می‌توان بلافاصله با قطعیت گفت که سمبل a ارسال شده است؟)

کدهایی که مشکل کد C_3 را نداشته باشند، کدهای «آنی» می‌نامیم.

۲. می‌توان نشان داد اگر هیچ‌یک از کلمات یک کد، پیشوند کلمه‌ی دیگری نباشد، کد آنی است و در نتیجه به طور یکتا قابل کشف و ناویژه هم خواهد بود. این کدها را کدهای «پیشوندی» می‌نامیم.

همچنین می‌توان نشان داد که شرط لازم و کافی برای وجود یک کد پیشوندی دودویی با طول کلمات l_1, l_2, \dots, l_M آنست که:

$$\sum_{i=1}^M 2^{-l_i} \leq 1$$

و از آنجا که:

$$\mathbb{E}[l(X)] = \sum_{i=1}^M p_i l_i$$

باید مسأله‌ی بهینه‌سازی زیر را حل کنیم:

$$\min_{l_1, l_2, \dots, l_M} \sum_{i=1}^M p_i l_i \quad \text{s.t.} \quad \sum_{i=1}^M 2^{-l_i} \leq 1$$

با استفاده از روش ضرایب لاگرانژ این مسأله را حل کنید و طول کلمه‌کدها را برای منبع X بیابید.

(راهنمایی: روش ضرایب لاگراژ با قید تساوی را در درس ریاضی عمومی ۲ آموخته‌اید. در اینجا قید به صورت نامساوی است، ولی دقت کنید که می‌خواهیم تا حدّ ممکن، l_i ها را کم کنیم و با کم کردن l_i ها، عبارت $\sum_{i=1}^M 2^{-l_i}$ بزرگ می‌شود. در نتیجه اگر فرض طبیعی بودن l_i ها را موقتاً کنار بگذاریم، نقطه‌ی بهینه در جایی رخ می‌دهد که قید به صورت تساوی برقرار شود.)

۳. به جواب نزدیک شده‌ایم، ولی هنوز یک قدم دیگر باقی مانده است!

با توجه به اینکه طول کلمه‌کدها را داریم و می‌خواهیم کد پیشوندی باشد، کلمه‌کدها را برای منبع X بسازید.

(راهنمایی: از کوچک‌ترین کلمه‌کد شروع کنید! توجه کنید که مسأله یک جواب ندارد و کفایت که یک جواب را گزارش کنید)

۴. طول متوسط کلمه‌کدها را بیابید.

۵. حال قصد داریم تصویر سیاه و سفید تولید کنیم و هر سمپل خروجی این منبع، عددی بین 0 تا 255 است (در نتیجه $M = 256$).

منبع اطلاعات زیر را در نظر بگیرید:

$$X = \begin{pmatrix} I_x = 0 & I_x = 50 & I_x = 100 & I_x = 150 & I_x = 200 & I_x = 250 & \text{otherwise} \\ \frac{1}{2} & \frac{1}{32} & \frac{1}{8} & \frac{1}{16} & \frac{1}{32} & \frac{1}{4} & 0 \end{pmatrix}$$

همانطور که می‌دانید، I_x میزان روشنایی پیکسل است که در بازه‌ای بین ۰ تا ۲۵۵ می‌تواند باشد و این طیف سیاه تا سفید را تشکیل می‌دهد. تابعی با عنوان تابعی به نام InformationSource بسازید که تصویری (آرایه دو بعدی) به اندازه‌ی 28×28 به صورت iid تولید کند که توزیع احتمالی مانند توزیع احتمال منبع X باشد.

۶. تابعی با عنوان SourceEncoder بنویسید که در ورودی تصویر ساخته شده در قسمت قبل را در ورودی دریافت کند و در خروجی رشته از صفر و یک‌ها بدهد که کدشده‌ی تصویر ورودی است.

۷. تابعی با عنوان SourceDecoder بنویسید که در ورودی، دنباله‌ای از صفر و یک‌ها دریافت کند و در خروجی، تصویر متناظر را بدهد.

۸. درستی توابع SourceEncoder و SourceDecoder را بررسی کنید. ۱۰ تصویر تصادفی تولید کنید و مراحل کدگذاری و کدگشایی را روی این تصاویر اعمال کنید. آیا تصاویر ورودی و خروجی برابر شد؟

۹. $n = 1000$ تصویر تصادفی تولید کنید. برای هر تصویر طول رشته کدگذاری شده را بیابید. برداری به طول n به نام L_avg بسازید و خانه‌ی i ام آن را میانگین طول رشته کد i عکس اول قرار دهید. نمودار تغییرات این بردار را رسم کنید. انتظار دارید میانگین خواسته شده به چه عددی میل کند؟

موفق باشید