Solutions to Problem Set 12

discussed:
29 July 2022

Data Compression With And Without Deep Probabilistic Models
Prof. Robert Bamler, University of Tuebingen

Course materials available at https://robamler.github.io/teaching/compress22/

Note: This problem set covers topics from the entire semester. The problems are designed to more closely resemble potential exam questions than some of the problems on previous problem sets.

Problem 12.1: Probabilities, Entropies, and Mutual Information

Consider an unbiased coin that, when tossed, comes up either "one" or "zero" with equal probability (and independent of any previous coin tosses). Assume that you flip the coin three times and let $X_i \in \{0, 1\}$ for $i \in \{1, 2, 3\}$ be the outcome of the i'th throw. Let $X_{\mathsf{sum}} = X_1 + X_2 + X_3$ be the total number of "one" throws.

In the following, all entropies should be calculated in *bits* (i.e., with base 2), as we always did in this course.

(a) What is the entropy $H_P(X_i)$ for each $i \in \{1, 2, 3\}$? (no calculation required; it suffices if you state the correct result.)

Solution: one bit ■

(b) What is the joint entropy $H_P\big((X_1, X_2, X_3)\big)$ of the tuple (X_1, X_2, X_3)? (no calculation required; it suffices if you state the correct result.)

Solution: three bits (independent random variables, so the entropies simply add up) ■

(c) What is the probability $P(X_{\mathsf{sum}} = x_{\mathsf{sum}})$ for each $x_{\mathsf{sum}} \in \{0, 1, 2, 3\}$?

Solution: Each of the $2^3 = 8$ combinations for $(X_1, X_2, X_3) \in \{0, 1\}^3$ has equal probability of $1/8$. Thus, $P(X_{\mathsf{sum}} = x_{\mathsf{sum}})$ is $1/8$ times the number of combinations $(X_1, X_2, X_3) \in \{0, 1\}^3$ that lead to the requested sum. We find: $P(X_{\mathsf{sum}} = 0) = 1/8$, $P(X_{\mathsf{sum}} = 1) = 3/8$, $P(X_{\mathsf{sum}} = 2) = 3/8$, and $P(X_{\mathsf{sum}} = 3) = 1/8$. ■

(d) What is the entropy $H_P(X_{\mathsf{sum}})$? Provide your result in the form $H_P(X_1) = a + b \log_2 3$ where a and b are rational numbers. Hint: $\log_2 8 = 3$.

Solution:

$$\begin{aligned} H_P(X_{\text{sum}}) &= -\frac{1}{8}\log_2\frac{1}{8} - \frac{3}{8}\log_2\frac{3}{8} - \frac{3}{8}\log_2\frac{1}{8} - \frac{1}{8}\log_2\frac{1}{8} \\ &= \frac{2}{8}\log_2 8 + \frac{6}{8}(\log_2 8 - \log_2 3) \\ &= 3 - \frac{6}{8}\log_2 3 \ (\approx 1.81) \end{aligned}$$

■

(e) What is the entropy $H_P\big((X_1, X_2, X_3, X_{\text{sum}})\big)$?

Solution: Since X_{sum} is a deterministic function of X_1, X_2, and X_3, it doesn't add any additional entropy:

$$H_P\big((X_1, X_2, X_3, X_{\text{sum}})\big) = \underbrace{H_P\big((X_1, X_2, X_3)\big)}_{=\,3\text{ (see part (b))}} + \underbrace{H_P\big(X_{\text{sum}}|X_1, X_2, X_3\big)}_{=0} = 3.$$

■

(f) What is the conditional probability $P(X_1{=}x_1 \,|\, X_{\text{sum}}{=}2)$ for each $x_1 \in \{0, 1\}$?

Solution: For $x_1 = 0$, we find:

$$P(X_1{=}0 \,|\, X_{\text{sum}}{=}2) = \frac{P(X_1{=}0, X_{\text{sum}}{=}2)}{P(X_{\text{sum}}{=}2)} = \frac{1/8}{3/8} = \frac{1}{3}.$$

Here, we used that $P(X_1{=}0, X_{\text{sum}}{=}2) = \frac{1}{8}$ because the two conditions $X_1{=}0$ and $X_{\text{sum}}{=}2$ are both satisfied only in the single configuration $(X_1, X_2, X_3) = (0, 1, 1)$, which occurs with probability $\frac{1}{8}$. For $x_1 = 1$, we find:

$$P(X_1{=}1 \,|\, X_{\text{sum}}{=}2) = 1 - P(X_1{=}0 \,|\, X_{\text{sum}}{=}2) = \frac{2}{3}.$$

■

(g) What is the conditional entropy $H_P(X_1 \,|\, X_{\text{sum}}{=}2)$? Provide your result again in the form $H_P(X_1) = a + b\log_2 3$ where a and b are rational numbers.

Solution: Using our result from Part (f), we find:

$$H_P(X_1 \,|\, X_{\text{sum}}{=}2) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} = \log_2 3 - \frac{2}{3}\log_2 2 = \log_2 3 - \frac{2}{3}.$$

■

(h) Let $X'_{\text{sum}} := X_1 + X_2$. Which of the following two statements about mutual informations is true?

(i) $I_P(X_1; X_{\text{sum}}) \geq I_P(X_1; X'_{\text{sum}})$; or

(ii) $I_P(X_1; X_{\text{sum}}) \leq I_P(X_1; X'_{\text{sum}})$.

(you may assume that only one of the two statements is true.) *Hint:* no calculation is needed, but write a brief statement (at most one sentence) to justify your answer.

Solution: $X_1 \to X'_{\text{sum}} \to X_{\text{sum}}$ forms a Markov chain (since $X_{\text{sum}} = X'_{\text{sum}} + X_3$), which implies that statement (ii) holds by the data processing inequality. ■

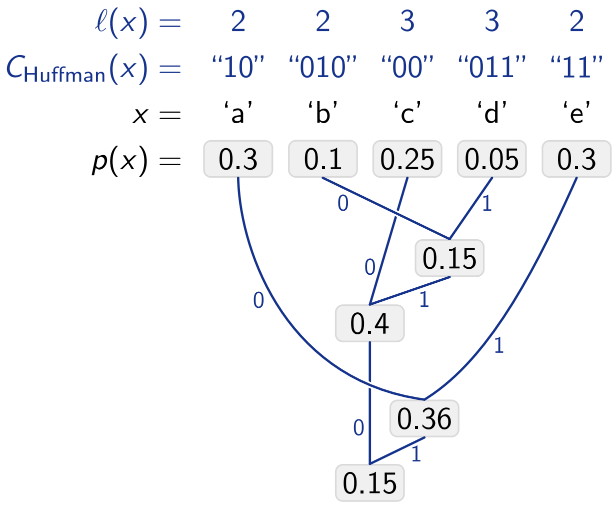
Problem 12.2: Source Coding Theorem & Huffman Coding

Consider the following probability distribution over a random variable X_i from the alphabet $\mathfrak{X} = \{\text{‘a’}, \text{‘b’}, \text{‘c’}, \text{‘d’}, \text{‘e’}\}$:

$$P(X_i = \text{‘a’}) = 0.3; \qquad P(X_i = \text{‘c’}) = 0.25; \qquad P(X_i = \text{‘e’}) = 0.3;$$
$$P(X_i = \text{‘b’}) = 0.1; \qquad P(X_i = \text{‘d’}) = 0.05.$$

(a) Draw a Huffman tree for this probability distribution and write down a table for a corresponding Huffman code $C_{\text{Huffman}} : \mathfrak{X} \to \{0,1\}^*$.

Solution:



$\ell(x) =$	2	2	3	3	2
$C_{\text{Huffman}}(x) =$	“10”	“010”	“00”	“011”	“11”
$x =$	‘a’	‘b’	‘c’	‘d’	‘e’
$p(x) =$	0.3	0.1	0.25	0.05	0.3

■

(b) Calculate the expected code word length, $L := \mathbb{E}_P\big[|C_{\text{Huffman}}(X_i)|\big]$ where $|\cdot|$ denotes the length of a bit string.

Solution:

$$L = 2 \times (0.3 + 0.1 + 0.3) + 3 \times (0.25 + 0.05) = 1.4 + 0.9 = 2.3$$

■

(c) Use your result for the expected code word length L and the source coding theorem to derive a lower and an upper bound for the entropy $H_P(X_i)$. Express your result in the form "a ? $H_P(X_i)$? $a + 1$" where a is a rational number and each "?" denotes either "$<$" or "$\leq$".

Solution: Huffman coding achieves the smallest possible expected code word length. According to the source coding theorem, the smallest possible expected code word length is *at least* the entropy $H_P(X_i)$ and *strictly less than* one bit more than the entropy. Thus,

$$H_P(X_i) \leq L < H_P(X_i) + 1 \quad \Rightarrow \quad L - 1 < H_P(X_i) \leq L.$$

■

Assume now that you have a random message $\mathbf{X} \in \mathfrak{X}^k$ consisting of k symbols $X_i \in \mathfrak{X}$, $i \in \{1, \ldots, k\}$ that are all statistically independent and distributed according to the above probability distribution. Assume that you encode these symbols with your Huffman code.

(a) What is the *expected* length $\mathbb{E}_P\big[|C_{\text{Huffman}}(\mathbf{X})|\big]$ of the encoded representation of the entire message $\mathbf{X}$?

Solution: Huffman codes are prefix free symbol codes. Thus, when encoding a sequence of symbols, we can simply concatenates their code words without any deliminators:

$$\mathbb{E}_P\big[|C_{\text{Huffman}}(\mathbf{X})|\big] = kL = k \times 2.3$$

■

(b) Let $\sigma^2 := \mathbb{E}_P\big[\big(|C_{\text{Huffman}}(X_i)| - L\big)^2\big]$ be the variance of the code word lengths in your Huffman code. Use the weak law of large number to provide a lower bound for the probability $P\big(\big||C_{\text{Huffman}}(\mathbf{X})|/k - L\big| < \beta\big)$ for arbitrary $\beta \geq 0$.

Solution: Since $|C_{\text{Huffman}}(\mathbf{X})| = \sum_{i=1}^{k} |C_{\text{Huffman}}(X_i)|$ is a sum of independent random variables with the same mean and (finite) variance, the weak law of large number applies:

$$P\big(\big||C_{\text{Huffman}}(\mathbf{X})|/k - L\big| < \beta\big) = 1 - P\big(\big||C_{\text{Huffman}}(\mathbf{X})|/k - L\big| \geq \beta\big) \geq 1 - \frac{\sigma^2}{k\beta^2}.$$

■

Problem 12.3: Variational Inference

Consider a so-called “hierarchical” latent variable model with observed data (or “message”) $\mathbf{X}$ and with *two* latent variables $\mathbf{Z}$ and $\mathbf{Z}'$. The joint probability distribution is:

$$P(\mathbf{Z}, \mathbf{Z}', \mathbf{X}) = P(\mathbf{Z})\, P(\mathbf{Z}'|\mathbf{Z})\, P(\mathbf{X}|\mathbf{Z}'). \tag{1}$$

Assume that we observe some data $\mathbf{x}$ and we now want approximate the (intractable) posterior distribution $P(\mathbf{Z}, \mathbf{Z}'|\mathbf{X} = \mathbf{x})$ with a variational distribution Q_ϕ (where ϕ are the variational parameters) that factorizes as follows,

$$Q_\phi(\mathbf{Z}, \mathbf{Z}' \,|\, \mathbf{X}{=}\mathbf{x}) = Q_\phi(\mathbf{Z} \,|\, \mathbf{X}{=}\mathbf{x})\, Q_\phi(\mathbf{Z}' \,|\, \mathbf{Z}, \mathbf{X}{=}\mathbf{x}). \tag{2}$$

To find the optimal variational parameters ϕ^*, we maximize the evidence lower bound (ELBO), which is defined analogous to our usual definition,

$$\mathrm{ELBO}(\phi) := \mathbb{E}_{Q_\phi(\mathbf{Z},\mathbf{Z}'|\mathbf{X}=\mathbf{x})}\left[\log \frac{P(\mathbf{Z}, \mathbf{Z}', \mathbf{X}{=}\mathbf{x})}{Q_\phi(\mathbf{Z}, \mathbf{Z}' \,|\, \mathbf{X}{=}\mathbf{x})}\right]. \tag{3}$$

Express the ELBO in the following form:

$$\mathrm{ELBO}(\phi) = -D_{\mathrm{KL}}\big(Q_\phi(\mathbf{Z} \,|\, \mathbf{X}{=}\mathbf{x}) \,\|\, P(\mathbf{Z})\big) - \mathbb{E}_{?}\big[D_{\mathrm{KL}}\big(? \,\|\, ?\big)\big] + \mathbb{E}_{Q_\phi(\mathbf{Z}'|\mathbf{X}=\mathbf{x})}\big[P(\mathbf{X}{=}\mathbf{x} \,|\, \mathbf{Z}')\big] \tag{4}$$

and fill in the three blanks marked with “?”. Here, D_{KL} is the Kullback-Leibler divergence.

Solution:

$$\begin{aligned}
\mathrm{ELBO}(\phi) &= \mathbb{E}_{Q_\phi(\mathbf{Z},\mathbf{Z}'|\mathbf{X}=\mathbf{x})}\left[\log \frac{P(\mathbf{Z}, \mathbf{Z}', \mathbf{X}{=}\mathbf{x})}{Q_\phi(\mathbf{Z}, \mathbf{Z}' \,|\, \mathbf{X}{=}\mathbf{x})}\right] \\
&= \mathbb{E}_{Q_\phi(\mathbf{Z}|\mathbf{X}=\mathbf{x})\, Q_\phi(\mathbf{Z}'|\mathbf{Z},\mathbf{X}=\mathbf{x})}\left[\log \frac{P(\mathbf{Z})\, P(\mathbf{Z}'|\mathbf{Z})\, P(\mathbf{X}|\mathbf{Z}')}{Q_\phi(\mathbf{Z} \,|\, \mathbf{X}{=}\mathbf{x})\, Q_\phi(\mathbf{Z}' \,|\, \mathbf{Z}, \mathbf{X}{=}\mathbf{x})}\right] \\
&= -\mathbb{E}_{Q_\phi(\mathbf{Z}|\mathbf{X}=\mathbf{x})}\left[\log \frac{Q_\phi(\mathbf{Z} \,|\, \mathbf{X}{=}\mathbf{x})}{P(\mathbf{Z})}\right] \\
&\quad - \mathbb{E}_{Q_\phi(\mathbf{Z}|\mathbf{X}=\mathbf{x})}\left[\mathbb{E}_{Q_\phi(\mathbf{Z}'|\mathbf{Z},\mathbf{X}=\mathbf{x})}\left[\log \frac{Q_\phi(\mathbf{Z}' \,|\, \mathbf{Z}, \mathbf{X}{=}\mathbf{x})}{P(\mathbf{Z}'|\mathbf{Z})}\right]\right] \\
&\quad + \mathbb{E}_{Q_\phi(\mathbf{Z}'|\mathbf{X}=\mathbf{x})}\big[P(\mathbf{X}{=}\mathbf{x} \,|\, \mathbf{Z}')\big] \\
&= -D_{\mathrm{KL}}\big(Q_\phi(\mathbf{Z} \,|\, \mathbf{X}{=}\mathbf{x}) \,\|\, P(\mathbf{Z})\big) \\
&\quad - \mathbb{E}_{Q_\phi(\mathbf{Z}|\mathbf{X}=\mathbf{x})}\big[D_{\mathrm{KL}}\big(Q_\phi(\mathbf{Z}' \,|\, \mathbf{Z}, \mathbf{X}{=}\mathbf{x}) \,\|\, P(\mathbf{Z}'|\mathbf{Z})\big)\big] \\
&\quad + \mathbb{E}_{Q_\phi(\mathbf{Z}'|\mathbf{X}=\mathbf{x})}\big[P(\mathbf{X}{=}\mathbf{x} \,|\, \mathbf{Z}')\big].
\end{aligned}$$

■

Problem 12.4: Bits-Back Coding And Asymmetric Numeral Systems (ANS)

Consider a latent variable model with latent variables $\mathbf{Z}$, observed data (or "message") $\mathbf{X}$, and joint probability distribution $P(\mathbf{Z}, \mathbf{X}) = P(\mathbf{Z})\, P(\mathbf{X}|\mathbf{Z})$.

(a) Assume you want to *encode* a message $\mathbf{x}$ using this latent variable model and the bits-back trick. This means that you have to follow three steps, where each step can be phrased in the following form:

$$\left\{\begin{matrix}\text{encode or}\\ \text{decode}\end{matrix}\right\} \left\{\begin{matrix}\mathbf{x} \text{ or}\\ \mathbf{z}\end{matrix}\right\} \text{ with entropy model } \left\{\begin{matrix}P(\mathbf{Z}) \text{ or}\\ P(\mathbf{X}\,|\,\mathbf{Z}{=}\mathbf{z}) \text{ or}\\ P(\mathbf{Z}\,|\,\mathbf{X}{=}\mathbf{x}).\end{matrix}\right\} \qquad (5)$$

Write down the three steps for *encoding* $\mathbf{x}$ in the correct order, phrasing each step in the form of Eq. 5.

Solution: The bits-back *encoder* follows these three steps:

1) decode $\mathbf{z}$ with entropy model $P(\mathbf{Z}\,|\,\mathbf{X}{=}\mathbf{x})$;
2) encode $\mathbf{x}$ with entropy model $P(\mathbf{X}\,|\,\mathbf{Z}{=}\mathbf{z})$;
3) encode $\mathbf{z}$ with entropy model $P(\mathbf{Z})$.

■

(b) Now formulate the three steps of the corresponding bits-back *decoder*, again phrasing each step in the form of Eq. 5.

Solution: The bits-back *decoder* inverts the three steps of the encoder, in reverse order:

1) decode $\mathbf{z}$ with entropy model $P(\mathbf{Z})$;
2) decode $\mathbf{x}$ with entropy model $P(\mathbf{X}\,|\,\mathbf{Z}{=}\mathbf{z})$;
3) encode $\mathbf{z}$ with entropy model $P(\mathbf{Z}\,|\,\mathbf{X}{=}\mathbf{x})$.

■

(c) As you've learned in the lecture, the Asymmetric Numeral Systems (ANS) algorithm can be understood as bits-back coding for each symbol X_i. Listing 1 shows the code for a simple (albeit slow) ANS coder. The code listing also contains a usage example just in case it is not clear what the coder implementation does. Consider the methods `push` and `pop` for encoding and decoding a symbol, respectively. For each of the three steps of the bits-back algorithm that you identified in parts (a) and (b) above, identify the (possibly empty) set of lines in the code listing that implements that step.

Solution: Following our convention from the lecture, we refer to the latent variable model that is used by ANS with the letter Q instead of P. Also, the message that we encode in a single call to `push` or `pop` is a single symbol, which we denote as X_i in the following.

- Method `push` (encoding a smbol):
  1) decode z_i with entropy model $Q(Z_i \,|\, X_i \!=\! x_i)$: lines 7-8;
  2) encode x_i with entropy model $Q(X_i \,|\, Z_i = z_i)$: not necessary here since the likelihood $Q(X_i \,|\, Z_i = z_i)$ is a deterministic probability distribution, i.e., the symbol x_i has zero information content under this likelihood;
  3) encode z_i with entropy model $Q(Z_i)$: line 9.
- Method `pop` (decoding a smbol):
  1) decode z_i with entropy model $Q(Z_i)$: lines 12-13.
  2) decode x_i with entropy model $Q(X_i \,|\, Z_i = z_i)$: not necessary here since the likelihood $Q(X_i \,|\, Z_i = z_i)$ is a deterministic probability distribution, i.e., we can identify x_i from z_i without any additional decoding (one could argue that this identification of x_i, which is implemented in lines 14-18, corresponds to step 2);
  3) encode z_i with entropy model $Q(Z_i \,|\, X_i \!=\! x_i)$: line 19;

■

Problem 12.5: Lossy Compression

Note: This is the only problem on the current problem set that would not be suitable for an exam question. This is done deliberately so that I can save a more self-contained problem for the exam.

In contrast to lossless compression, lossy compression can also be used for *continuous* data. Interestingly, the rate/distortion-theorem holds for continuos data in the same way as it does for discrete data, i.e., the optimal expected amortized bit rate for a given lossy compression is given by the mutual information $I_P(X; \hat{X})$ between the original message X and the reconstruction $\hat{X}$.

Consider a data source that generates scalar continuous messages $X \in \mathbb{R}$ with distribution $P(X) = \mathcal{N}(X; 0, \sigma^2)$, i.e., normal distributed with zero mean and standard deviation σ^2. After encoding and decoding, the reconstructed symbol $\hat{X} \in \mathbb{R}$ acquires some additional Gaussian noise with variance γ^2, i.e., $P(\hat{X}|X) = \mathcal{N}(\hat{X}; X, \gamma^2)$, i.e., given X, the reconstruction $\hat{X}$ is normal distributed with mean X and variance γ^2. Thus, the marginal distribution of the reconstruction, $P(\hat{X}) = \mathbb{E}_{P(X)}[P(\hat{X}|X)]$ is the convolution of two normal distributions. It is well-known that this convolution results again in a Gaussian, where the variances add up, i.e., $P(\hat{X}) = \mathcal{N}(\hat{X}; 0, \sigma^2 + \gamma^2)$ (this is known as "Gaussian error propagation").

(a) The mutual information for continuous variables is defined as follows,

$$I_P(X;\hat{X}) = \mathbb{E}_P\left[\log \frac{p(X,\hat{X})}{p(X)\,p(\hat{X})}\right] \tag{6}$$

where lower case "p" denotes the probability density function. Convince yourself that, analogous to the case of discrete random variables, one can equivalently express $I_P(X;\hat{X})$ as follows,

$$I_P(X;\hat{X}) = h_P(\hat{X}) - h_P(\hat{X}|X) \tag{7}$$

where lower case h_P denotes the *differential* entropy $h_p(X) := -\int p(x)\log_2 p(x)\,dx$ and $h_P(\hat{X}|X) := -\iint p(x,\hat{x})\log_2 p(\hat{x}|x)\,d\hat{x}\,dx$ with $p(\hat{x}|x) := p(x,\hat{x})/p(x)$.

Note: while a differential entropy can be negative, one can show that the difference of differential entropies on the right-hand side of Eq. 7 is always positive.

Solution: Eq. 7 follows directly from Eq. 6 when we identify $p(X,\hat{X})/p(X)$ as the conditional probability density function $p(\hat{X}|X)$ and then express the expectation over P as an integral and use the marginalization $\int p(x,\hat{x})\,dx = p(\hat{x})$:

$$\begin{aligned} I_P(X;\hat{X}) &= \mathbb{E}_P\left[\log \frac{p(\hat{X}|X)}{p(\hat{X})}\right] = \iint p(x,\hat{x})\log_2 \frac{p(\hat{x}|x)}{p(\hat{x})}\,dx\,d\hat{x} \\ &= h_P(\hat{X}) - h_p(\hat{X}|X). \end{aligned}$$

■

(b) Look up the differential entropy of a normal distribution on Wikipedia (it is simply called "entropy" there) and calculate the mutual information (and thus the optimal expected amortized bit rate) $I_P(X;\hat{X})$ using Eq. 7. Express your result as a function of the *signal to noise ration* σ^2/γ^2.

Solution: Wikipedia gives the differential entropy of a normal distribution with variance σ^2 as $\frac{1}{2}\log(2\pi\sigma^2) + \frac{1}{2}$. Note that Wikipedia follows the convention to measure entropies with base e and, accordingly, the "log" denotes here the natural logarithm. In the field of data compression, it is often more useful to measure entropies in base 2 so that they correspond to practical bit rates. To translate from base e to base 2, we first reformulate $\frac{1}{2}\log(2\pi\sigma^2) + \frac{1}{2} = \frac{1}{2}\log(2\pi e\sigma^2)$ and then change the base of the logarithm to 2, resulting in the entropy

$$h_P(X) = \frac{1}{2}\log_2(2\pi e\sigma^2).$$

To evaluate the right-hand side of Eq. 7, we now need:

- the entropy of the marginal distribution $P(\hat{X})$ of the reconstruction; as noted above, we have $P(\hat{X}) = \mathcal{N}(\hat{X};0,\sigma^2+\gamma^2)$ and thus we find $h_P(\hat{X}) = \frac{1}{2}\log_2\big(2\pi e(\sigma^2+\gamma^2)\big)$.

- the entropy of the conditional distribution $P(\hat{X}|X) = \mathcal{N}(\hat{X}; X, \gamma^2)$; since the entropy of a normal distribution only depends on the variance and not on the mean, $h_P(\hat{X}|X)$ is independent of X and we have $h_P(\hat{X}|X) = \frac{1}{2}\log_2(2\pi e \gamma^2)$.

Using Eq. 7, we therefore find

$$\begin{aligned} I_P(X;\hat{X}) = h_P(\hat{X}) - h_P(\hat{X}|X) &= \frac{1}{2}\log_2\big(2\pi e(\sigma^2+\gamma^2)\big) - \frac{1}{2}\log_2(2\pi e\gamma^2) \\ &= \frac{1}{2}\log_2\left(1+\frac{\sigma^2}{\gamma^2}\right). \end{aligned}$$

Thus, the minimal required bitrate for lossy compression of continuous data increases with increasing signal-to-noise ration σ^2/γ^2, as expected.

Note: it is instructive to consider the two limits of very low and very high signal-to-noise ratio:

- For very high signal-to-noise ratio $\frac{\sigma^2}{\gamma^2} \gg 1$, we can neglect the "1+" inside the logarithm and obtain

  $$I_P(X;\hat{X}) \approx \frac{1}{2}\log_2\frac{\sigma^2}{\gamma^2} = \log_2\frac{\sigma}{\gamma} \qquad \left(\text{for } \frac{\sigma^2}{\gamma^2} \gg 1\right).$$

  i.e., the optimal bit rate grows logarithmically with the signal-to noise ration. This result seems plausible since a logarithmically increasing bit rate admits for a linearly increasing range of values that can be encoded.

  Recall that we're considering here a setup where we have a signal that is normal distributed with variance σ^2, and the lossy compression distorts the signal by some additive noise with variance γ^2. Thus, *loosely speaking*, most signals are roughly from the interval $[-\sigma, \sigma]$, and the noise with amplitude γ prevents us from distinguishing signals that are closer than γ to each other. Thus, our setup is similar to dividing the signal space $[-\sigma, \sigma]$ into a grid with spacing γ, resulting in $\propto \sigma/\gamma$ grid points. Enumerating these grid points would require on the order of $\log_2(\sigma/\gamma)$ bits, which is exactly what we obtain in the regime of high signal-to-noise ratio.

- For very low signal-to-noise ratio $\frac{\sigma^2}{\gamma^2} \ll 1$, the above intuition breaks down because we cannot divide the interval $[-\sigma, \sigma]$ into a grid with spacing γ if the grid spacing is larger than the interval itself. In this limit, we can use the Taylor approximation $\log_2(1+\epsilon) \approx \epsilon/\ln 2$ for small ϵ. Thus, we find:

  $$I_P(X;\hat{X}) \approx \frac{1}{2\ln 2}\frac{\sigma^2}{\gamma^2} \qquad \left(\text{for } \frac{\sigma^2}{\gamma^2} \ll 1\right).$$

  i.e., the optimal bit rate is approximately proportional to the signal-to-noise ratio $\frac{\sigma^2}{\gamma^2}$.

■

```python
class SimpleAnsCoder:
    def __init__(self, precision, compressed=0):
        self.n = 2**precision         # ("**" denotes exponentiation)
        self.compressed = compressed

    def push(self, symbol, m):        # Encodes one symbol.
        z = self.compressed % m[symbol] + sum(m[0:symbol])
        self.compressed //= m[symbol] # ("//" denotes integer division)
        self.compressed = self.compressed * self.n + z

    def pop(self, m):                 # Decodes one symbol.
        z = self.compressed % self.n
        self.compressed //= self.n    # ("//" denotes integer division)
        for symbol, m_symbol in enumerate(m):
            if z >= m_symbol:
                z -= m_symbol
            else:
                break
        self.compressed = self.compressed * m_symbol + z
        return symbol

    def get_compressed(self):
        return self.compressed

# USAGE EXAMPLE:

# Define an approximate entropy model Q with 4 bits of precision and
# $Q_i(X_i{=}0) = \frac{7}{2^4}$, $Q_i(X_i{=}1) = \frac{3}{2^4}$, and $Q_i(X_i{=}2) = \frac{6}{2^4}$.
precision = 4
m = [7, 3, 6]

# Encode an example message (in reversed order):
example_message = [2, 0, 2, 1, 0]
encoder = SimpleAnsCoder(precision)
for symbol in reversed(example_message):
    encoder.push(symbol, m) # We could use a different m for each symbol.
compressed = encoder.get_compressed()
print(f'Compressed bit string: {compressed:b}')

# Decode the example message:
decoder = SimpleAnsCoder(precision, compressed)
reconstructed = [decoder.pop(m) for _ in range(5)]
assert reconstructed == example_message # Verify correctness.
```

Listing 1: A simple (but slow) ANS coder.