MPRI 2.3.2, Foundations of Privacy
Exercises

It is hightly recommended to try to solve the exercises before looking at the answers. In the exam answers don't have to be long: sort and to the point answers are enough, but the have to provide clear and sufficient arguments. In the answers below, places where the answer needs further expansion are mentioned.

1 Quantitative Information Flow

Exercise 1

Let C be a channel from $\mathcal{X}$ to $\mathcal{Y}$.

1 Show that for any prior π and gain function g:

$$\mathcal{L}_g^\times(\pi, C) \le |\mathcal{Y}| \qquad \text{and}$$
$$\mathcal{L}_g^\times(\pi, C) \le |\mathcal{X}|$$

2 Let π_u be the uniform prior. Show that

$$(\forall g : \mathcal{L}_g^\times(\pi_u, C) = 1)$$

if and only if C is non-interfering.

Answer Recall that the (multiplicative) leakage for any π, g is bounded by the Bayes-capacity, which has a simple expression: it is given by the sum of the column maxima. That is:

$$\mathcal{L}_g^\times(\pi, C) \le \mathcal{ML}^\times(C) = \textstyle\sum_y \max_x C_{xy}$$

1 $\sum_y \max_x C_{xy} \le |\mathcal{Y}|$ holds because $C_{xy} \le 1$ for all x, y (they're probabilities).

Moreover, the sum of each row of C is 1 (each row is a probability distribution), hence the sum of all elements of C is $|\mathcal{X}|$. The expression of Bayes capacity sums a subset of all elements, so it has to be vounded by $|\mathcal{X}|$. Concretely: $\sum_y \max_x C_{xy} \le \sum_y \sum_x C_{xy} = |\mathcal{X}|$.

2 Recall that C is non-interfering iff all secrets x, x' produce each y with the same probability, i.e. $C_{xy} = C_{x'y}$.

Assuming non-interference, $\max_x C_{xy}$ is always equal to C_{x^*y} for some fixed secret x^*, so the Bayes-capacity is $\sum_y \max_x C_{xy} = \sum_y C_{x^*y} = 1$, and $\mathcal{L}_g^\times(\pi_u, C)$ is always bounded by it.

Assuming $(\forall g : \mathcal{L}_g^\times(\pi_u, C) = 1)$, we know that Bayes-capacity must be 1 since it's equal to $\mathcal{L}_{g_{id}}^\times(\pi_u, C)$ where g_{id} is the identity gain function. With a similar reasoning we conclude that $\max_x C_{xy}$ has to be equal to C_{x^*y} for any fixed x^* which implies non-interference.

Exercise 2

1 Consider an instance of the Dining Cryptographers protocol with 3 cryptographers on a *line*:

$$\mathrm{Crypt}_1 \text{ —— } \mathrm{Crypt}_2 \text{ —— } \mathrm{Crypt}_3$$

That is, there is a coin between $\mathrm{Crypt}_1/\mathrm{Crypt}_2$ and $\mathrm{Crypt}_2/\mathrm{Crypt}_3$, but not between $\mathrm{Crypt}_1/\mathrm{Crypt}_3$.

Model the system as a channel. Is it non-interfering? Compute its multiplicative Bayes-capacity.

2 Now consider the usual instance of 3 Dining Cryptographers on a ring, but assume that the coin shared between $\mathrm{Crypt}_1/\mathrm{Crypt}_3$ is *observable* (i.e. visible to the adversary).

Repeat the question (1) for this variant.

3 Can we avoid computing the multiplicative Bayes-capacity in question (2) directly, but obtain it by comparing the channel of question (2) with that of question (1)?

Answer

1 Assume that the bit to be sent is 1. Similarly to the case of a ring shown in the slides, there are 3 secrets and 4 observations $001, 010, 100, 111$ (written in the same order as the cryptographers). A case-analysis of the 2 coins needs to be done (for every possible sender), showing that $C_{xy} = 1/4$ for all cases, exactly like the case of a ring (Expand). This is a non-interfering channel hence its Bayes-capacity must be 1.

2 This makes the observations more informative, y's are now of the form (announcement, coin). All previous announcements are still possible, with both values of the visible coin, i.e.:

$$(001, 0) \quad (001, 1) \quad (010, 0) \quad (010, 1) \quad (100, 0) \quad (100, 1) \quad (111, 0) \quad (111, 1)$$

A case analysis of the two non-visible coins needs to be done (for every possible sender), showing that $C_{xy} = 1/8$ in all cases (Expand). So it's still a non-interfering channel, hence its Bayes-capacity must be one.

3 Let C^1, C^2 be the channels of questions (1),(2) respectively. The main observation here is that C^2 can be obtained from C^1 using a suitable (probabilistic) post-processing, i.e. $C^2 = C^1 R$ for a suitable channel R. Intuitively, C^1 provides the output of the system when there is no coin between Crypt_1 and Crypt_3. In question (2) such a coin exists, so in the post-processing we can select its value with uniform probability, then update the value of Crypt_1,Crypt_3 by adding the value of the visible coin, and output the updated announcements plus the value of the visible coin.

More precisely R is a channel with inputs of the form 001 and outputs of the form $001, 1$.

$$\begin{array}{cc}
 & \begin{array}{cccccccc} (001,0) & (001,1) & (010,0) & (010,1) & (100,0) & (100,1) & (111,0) & (111,1) \end{array} \\
\begin{array}{c} 001 \\ 010 \\ 100 \\ 111 \end{array} & \left[\begin{array}{cccccccc}
1/2 & 0 & 0 & 0 & 0 & 1/2 & 0 & 0 \\
0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 1/2 \\
0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/2 & 0 & 0 & 1/2 & 0
\end{array}\right]
\end{array}$$

In each row a probabilistic choice is made (for the visible coin), with probability 1/2 the coin lands 0 and the announcements stay the same, and with probability 1/2 the coin lands 1 and Crypt_1,Crypt_3 flip their announcement.

From $C^2 = C^1 R$ we get that $C^2 \sqsubseteq_\circ C^1$, hence C^2's Bayes-capacity is less than C^1's, but C^1 has capacity 1 (the minumum possible) so C^2 Bayes-capacity must be also 1.

Exercise 3

In the Crowds protocol, due to the probabilistic routing, each request could pass through *corrupted* users *multiple times* before arriving to the server, as shown in the figure below. However, in the security analysis, we only considered as "detected" the *first* user who forwards the request to a corrupted one.

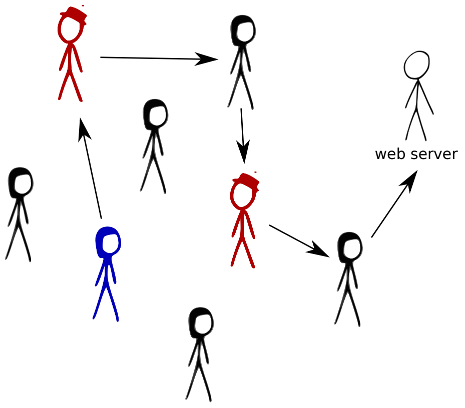
To perform a more precise analysis, let us consider the first *two* detected users, instead of only the first one. Let n, m be the number of honest and total users respectively. The set of secrets is still $\mathcal{X} = \{1, \ldots, n\}$ (we are only interested in the privacy of honest users).

On the other hand, the information available to the adversary is now more detailed. Observations are of the form $y = (d_1, d_2)$ where $d_1 \in \{1, \ldots, n, \perp\}$ (the first detected user, similarly to the original analysis) and $d_2 \in \{1, \ldots, m, \perp\}$ (the second detected user, who might be corrupted himself).

Show that this extra information is in fact useless to the adversary. More precisely, show that for any prior π and gain function g:

$$V_g(\pi, C^1) = V_g(\pi, C^2)$$

where C^2 is the channel obtained by the detailed analysis, considering two detected users, and C^1 is the channel of the original analysis, considering a single detected user.



Answer The easiest way of showing $V_g(\pi, C^1) = V_g(\pi, C^2)$ for all π, g is to show that C^1, C^2 are composition-refinements of each other, i.e.

$$C^1 \sqsubseteq C^2 \qquad \text{and} \qquad C^2 \sqsubseteq C^1$$

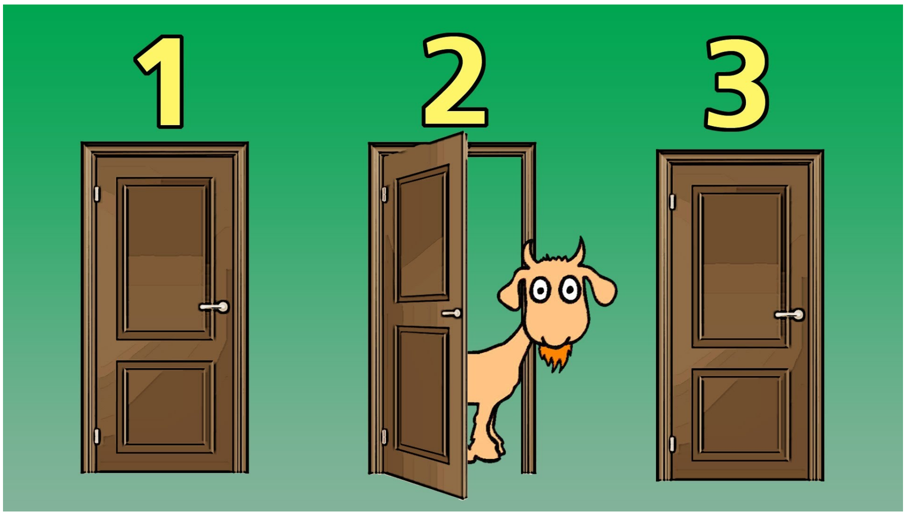
So we need to post-process C^1 into C^2 and vice-versa.

The one direction is easy: if we have $y = (d_1, d_2)$ we simply need to "forget" d_2 and keep only d_1 (expand the definition of this post-processing channel).

The other direction is a bit more involved. Starting from d_1 we need to construct d_2. The key observation is that, after the first detection, the message is now at the possesion of a corrputed user, and the route from this point on *does not depend* on the initial sender. So we just need to finish the protocol from the first detection until the end. First we select randomly one of the $c = m - n$ corrupted users to continue the route from there (the first detection happened by any one of them with equal probability). Then we select whether to forwaward (prob. p_f) or not (prob. $1 - p_f$), and if we select to forward essentially a new instance of Crowds starts, only with m instead of n users.

The values of the post-processing channel can be computed as a function of the values α, β, γ of the original Crowds channel (expand, but no need to compute actual numbers).

Exercise 4



The Monty Hall problem

You are presented with three doors: one contains a *price*, the two others a *goat*. You choose one of the doors, and then the host (who knows which door contains the price), opens one *goat* door among the two that were not chosen. (he never opens the price, always a goat among the two non-chosen ones).

You now have two options:

- *keep* your original choice, or
- *change it* for the other closed door.

What should you do?

This is a truly great puzzle, if you never heard of it before don't spoil the answer, think about it! Questions are in the next page.

1 Answer the question using solely *Quantitative Information Flow* concepts

2 What if we *only know which door was opened*, not the player's choice?

3 What if the host *does not know where the price is*, he just happened to open a goat?

Answer

1 The secret information in this problem is clearly the door containing the price. After the first phase of the problem we learn two pieces of information: the door that the player chooses and the door that the host opens. The problem can be then modelled by a channel with input $\mathcal{X} = \{1,2,3\}$ (the door with the price) and outputs $\mathcal{Y} = \{(1,2),(1,3),(2,1),(2,3),(3,1),(3,2)\}$ where $(1,2)$ means the player selected door 1 and the host opened door 2 (note that the host never opens the door selected by the player).

To construct the channel, let's assume that the player's choise is made uniformly and that, if the host has two doors with a goat that he can open - i.e. the player chose the right one - he also picks one uniformly. Hence the channel is the following one:

$$\begin{array}{c|cccccc} & (1,2) & (1,3) & (2,1) & (2,3) & (3,1) & (3,2) \\ \hline 1 & 1/6 & 1/6 & 0 & 1/3 & 0 & 1/3 \\ 2 & 0 & 1/3 & 1/6 & 1/6 & 1/3 & 0 \\ 3 & 1/3 & 0 & 1/3 & 0 & 1/6 & 1/6 \end{array}$$

Note that some probabilities are 0 because the host never opens the correct door. And some are 1/6 because the player chose the correct door so the host has two possible goats to choose from.

Now recall that Bayes vulnerability gives the adversary's (in this case the player) probability of guessing correctly, and the Bays-leakage compares his prior probability (before playing the game) with the posterior one (after completing the first phase). Assuming a uniform prior (the selection of the door where the price is put is uniform) the prior Bayes-vulnerability is 1/3.

The Bays-leakage, for a uniform prior, coincides with the Bayes-capacity, which is given by the sum of the column maxima, in this case $1/3 \cdot 6 = 2$. Hence the ratio between prior and posterior vulnerability is 2, which means that the posterior vulnerability is $2 \cdot 1/3 = 2/3$: after the first phase, the player has now $2/3$ chances of a correct guess (given by a strategy that changes his choice for the other door)!

This might sound counter-intuitive (and it is what makes the puzzle famous) but its true, and the above reasoning provides a proof.

2 If we only know the door opened by the host, not the player's choice, the channel becomes as follows:

$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 1/2 & 1/2 \\ 2 & 1/2 & 0 & 1/2 \\ 3 & 1/2 & 1/2 & 0 \end{array}$$

The Bayes-capacity is now $3 \cdot 1/2 = 3/2$, hence the posterior Bayes-vulnerability is $3/2 \cdot 1/3 = 1/2$. Which means that after the first phase the player has probability $1/2$ of a correct guess (by uniformly picking one of the remaining two doors).

Intuitively, in the standard case, when the player choses wrong (i.e. 2 out of 3 times), the host has no choice, he has to pick the only available goat, effectively revealing the location of the price (the "third door", selected by neither the player nor the host). But if we don't know the player's choice we have strictly less information, we don't know which one is the "third door".

3 If the host does not know where the price is, but just happened to open a goat, we need to start from a distribution where all 6 possible outcomes have equal probability, and then *condition on the event* that the host did not choose the price.

$$\begin{array}{c|cccccc} & (1,2) & (1,3) & (2,1) & (2,3) & (3,1) & (3,2) \\ \hline 1 & 1/4 & 1/4 & 0 & 1/4 & 0 & 1/4 \\ 2 & 0 & 1/4 & 1/4 & 1/4 & 1/4 & 0 \\ 3 & 1/4 & 0 & 1/4 & 0 & 1/4 & 1/4 \end{array}$$

Now all possible outcomes in each row have equal probability, and the capacity is $6 \cdot 1/4 = 3/2$. So, similarly to the previous case, the player has probability $1/2$ to guess correctly. In this case the host is not forced to reveal the price by showing the only available goat, he just happened to do so.

Exercise 5 Consider two programs on *uniformly distributed 64-bit integers*

- C_1: if $x \,\%\, 8 == 0$ then $y = x$ else $y = 1$
  completely reveals x one-eighth of the time
- C_2: $y = x \mid 00\ldots0111$
  always reveals all but the last three bits of x

1 Show that both channels have the same Bayes-leakage

2 Consider an adversary that has 3 tries to guess the secret. Model such an adversary by a suitable gain function and show that C_2 leaks more than C_1 under that gain function.

2 Utility

The utility of an oblivious mechanism, mapping query outcomes $\mathcal{Y}$ to reported values $\mathcal{Z}$, under a gain function g with set of guesses $\mathcal{W}$, is given by:

$$\mathcal{U} = \sum_{z\in\mathcal{Z}} \max_{w\in\mathcal{W}} \sum_{y\in\mathcal{Y}} p(y)p(z|y)g(w,y) \tag{1}$$

Note that an oblivious mechanism can be seen as a channel (in the sense of Quantitative Information Flow) from $\mathcal{Y}$ to $\mathcal{Z}$, and the formula above is exactly the posterior g-vulnerability of this channel, taking as prior the probability distribution $p(y)$ over the query outcomes.

For the "identity" gain function, having $\mathcal{W} = \mathcal{Y}$ as the set of guesses, and defined as $g(w,y) = 1$ if $w = y$ and 0 otherwise, the above formula can be simplified. We notice that $p(y)p(z|y)g(w,y) = p(w)p(z|w)$ if $y = w$ and 0 otherwise, so the formula becomes:

$$\mathcal{U} = \sum_{z\in\mathcal{Z}} \max_{y\in\mathcal{Y}} p(y)p(z|y) \tag{2}$$

(which is the posterior Bayes-vulnerability of the channel).

1. Compute the utility of the geometric mechanism for a counting query, with privacy degree ϵ, on the uniform prior distribution, with the gain function defined as the identity relation

Let n be the number of users, the result of a counting query is between 0 and n, hence $\mathcal{Y} = \{0,\ldots,n\}$.

The uniform distribution could be considered either as the distribution of the query outcomes (i.e. $p(y)$), or as the distribution of the users' value (in which case $p(y)$ becomes a binomial distribution). For simplicity we consider here $p(y)$ to be uniform, that is $p(y) = \frac{1}{n+1}$.

The geometric mechanism can output any integer (i.e. $\mathcal{Z} = \mathbb{Z}$), and for a counting query (i.e. for $\Delta_f = 1$) it is given by

$$p(z|y) = c\alpha^{|z-y|} \quad \text{where } \alpha = e^{-\epsilon}, c = \frac{1-\alpha}{1+\alpha}$$

Hence the utility under the identity gain function, given by (2), becomes:

$$\mathcal{U} = \frac{c}{n+1} \sum_{z\in\mathbb{Z}} \max_{y\in\mathcal{Y}} \alpha^{|z-y|} \tag{3}$$

Now let's consider the quantity $\max_{y\in\mathcal{Y}} \alpha^{|z-y|}$ for different values of z. Since $\alpha < 1$, the maximum is given by the y that minimizes $|z-y|$, i.e. the y that is closer to z. So for $0 \le z \le n$ we pick $y = z$, for $z \le 0$ we pick $y = 0$ and for $z \ge n$ we pick $y = n$. Hence we have:

$$\max_{y\in\mathcal{Y}} \alpha^{|z-y|} = \begin{cases} \alpha^{|z-z|} = 1 & \text{if } 0 \le z \le n \\ \alpha^{|z-0|} & \text{if } z \le 0 \\ \alpha^{|z-n|} & \text{if } z \ge n \end{cases}$$

So we can expand the sum of (3):

$$\begin{aligned} \sum_{z\in\mathbb{Z}} \max_{y\in\mathcal{Y}} \alpha^{|z-y|} &= \sum_{z=-\infty}^{0} \alpha^{|z|} + \sum_{z=1}^{n-1} 1 + \sum_{z=n}^{\infty} \alpha^{|z-n|} \\ &= n - 1 + 2\sum_{d=0}^{\infty} \alpha^d \\ &= n - 1 + 2\frac{1}{1-\alpha} && \text{(geometric series for } \alpha < 1) \end{aligned}$$

so from (3) we finally get:

$$\begin{aligned}
\mathcal{U} &= \frac{c}{n+1}(n + \frac{2}{1-\alpha} - 1) \\
&= \frac{c}{n+1}(n + \frac{2}{1-\alpha} - \frac{1-\alpha}{1-\alpha}) \\
&= \frac{c}{n+1}(n + \frac{1}{c}) \\
&= \frac{cn+1}{n+1}
\end{aligned}$$

It's worth taking a look at this quantity as a function of c. The utility under the identity gain function is simply the probability to correctly guess the query outcome (that is, the posterior Bayes-vulnerability). Recall that

$$c = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}}$$

Since $c \leq 1$ we have $\mathcal{U} \leq 1$, as expected (it's a probability).

Consider the one extreme case: perfect privacy. The geometric mechanism is well-defined for $\epsilon > 0$, but as $\epsilon \to 0$ the noise increases and $p(z|y), p(z|y'), y \neq y'$ become closer to each other. At the limit we have $c = 0$ hence $\mathcal{U} = \frac{1}{n+1}$. Intuitively, the output of the mechanism is useless, we still have $\frac{1}{n+1}$ probability of guessing the correct value.

At the other extreme case (no privacy at all), when $\epsilon \to \infty$ then $p(\cdot|y)$ becomes a point distribution, giving $p(y|y) = 1$ and $p(z|y) = 0$ for $z \neq y$. At the limit c becomes 1, hence $\mathcal{U} = 1$: as expected we can now guess the correct value with probability 1.

2. Same exercise, but with the gain function defined as the converse of the distance.

The goal here is not to come up with an exact closed-form formula, but to see how the utility changes because of the gain function. Under this gain function we still have $\mathcal{W} = \mathcal{Y}$ but the gain is given by

$$g(w, y) = n - |y - w|$$

The closer our guess w is to the real answer y, the higher the gain.

From (1) we get

$$\begin{aligned}
\mathcal{U} &= \sum_{z\in\mathcal{Z}} \max_{w\in\mathcal{W}} \sum_{y\in\mathcal{Y}} p(y)p(z|y)(n - |y-w|) \\
&= n - \sum_{z\in\mathcal{Z}} \min_{w\in\mathcal{W}} \sum_{y\in\mathcal{Y}} p(y)p(z|y)|y-w| \\
&= n - \frac{c}{n+1} \sum_{z\in\mathcal{Z}} \min_{w\in\mathcal{W}} \sum_{y\in\mathcal{Y}} \alpha^{|y-z|}|y-w|
\end{aligned}$$

We need to investigate the value w that gives the minimum:

$$\sum_{y\in\mathcal{Y}} \alpha^{|y-z|}|y-w|$$

for each z. To minimize this quantity we need that the factor $|y-w|$ is as small as possible when $\alpha^{|y-z|}$ is big, which happens if w is as close as possible to z. Hence, similarly to the previous case, for $0 \leq z \leq n$ we pick $w = z$, for $z \leq 0$ we pick $w = 0$ and for $z \geq n$ we pick $w = n$.

Finally we can expand the formula of utility to:

$$\mathcal{U} = n - \frac{c}{n+1}(\sum_{z=-\infty}^{-1} \sum_{y=0}^{n} \alpha^{y-z} y + \sum_{z=0}^{n}\sum_{y=0}^{n} \alpha^{|y-z|}|y-z| + \sum_{z=n+1}^{\infty} \sum_{y=0}^{n} \alpha^{z-y}(n-y))$$

3. Find a mechanism for the same counting query, with the same degree of privacy, but lower utility

Intuitively, the geometric mechanism is optimal because the noise is exactly as much as required by ϵ, not more. That is, the constraints of differential privacy for adjacent y's are satisfied with equality:

$$p(z|y) = e^{\epsilon} p(z|y+1) \quad 0 \leq y < n, z \in \mathbb{Z}$$

To degrade its utility, we could, for instance, add more noise to a certain query outcome. For instance, for $y = 0$ we can use the same distribution that we use for $y = 1$ (i.e. $p(z|0) = p(z|1)$). This modified version of the geometric mechanism is given by:

$$p(z|y) = c\alpha^{|z-\max\{1,y\}|}$$

That is, $p(z|0) = p(z|1) = c\alpha^{|z-1|}$ and $p(z|y) = c\alpha^{|z-y|}$ for $y > 1$.

Overall, the mechanism still satisfies differential privacy for the same ϵ, since the constraints for $p(z|1)$ and $(z|2)$ are still matched with equality. For any $\epsilon' < \epsilon$ we would have $p(z|1) > e^{\epsilon'} p(z|2)$.

On the other hand, utility is now lower, because we cannot distinguish 0 from 1: any value $z \leq 1$ should be mapped to either 0 or 1. Redoing the computation of the first exercise, we have:

$$\max_{y \in \mathcal{Y}} \alpha^{|z-\max\{1,y\}|} = \begin{cases} \alpha^{|z-z|} = 1 & \text{if } 1 \leq z \leq n \\ \alpha^{|z-1|} & \text{if } z \leq 1 \\ \alpha^{|z-n|} & \text{if } z \geq n \end{cases}$$

So the sum of (3) becomes:

$$\begin{aligned} \sum_{z \in \mathbb{Z}} \max_{y \in \mathcal{Y}} \alpha^{|z-y|} &= \sum_{z=-\infty}^{1} \alpha^{|z-1|} + \sum_{z=2}^{n-1} 1 + \sum_{z=n}^{\infty} \alpha^{|z-n|} \\ &= n - 2 + 2\sum_{d=0}^{\infty} \alpha^{d} \\ &= n - 2 + 2\frac{1}{1-\alpha} && \text{(geometric series for } \alpha < 1\text{)} \end{aligned}$$

and continuing similarly to the exercise 1, we get

$$\mathcal{U} = \frac{c(n-1)+1}{n+1}$$

Even under no privacy, when $\epsilon \to \infty$ and $c \to 1$, we have $\mathcal{U} = \frac{n}{n+1}$ (compared to $\mathcal{U} = 1$ for the original geometric mechanism), since two out of the $n+1$ elements are still completely indistinguishable!

4. We saw that post-processing cannot decrease privacy. Can it decrease the utility? Motivate your answer

Post-processing can create more confusion between the reported values. This does not decrease privacy (it can only become harder to infer the value of an individual) but it can decrease utility (it also becomes harder to infer the real outcome of the query).

A trivial example would be a constant post-processing function mapping every z to 0. The result of applying this post-processing is a non-interferent channel: it outputs 0 independently from the 0. This has perfect privacy but clearly no utility at all.