`fn sample_discrete_laplace`

Michael Shoemate

September 16, 2025

This proof resides in **"contrib"** because it has not completed the vetting process.

Proves soundness of `fn sample_discrete_laplace` in `mod.rs` at commit 0be3ab3e6 (outdated[1]). This proof is an adaptation of subsection 5.2 of [CKS20].

Vetting history

- Pull Request #519
- Pull Request #1134

1 Hoare Triple

Precondition

$\texttt{scale} \in \mathbb{Q} \wedge \texttt{scale} > 0$

Pseudocode

```python
def sample_discrete_laplace(scale) -> int:
    if scale == 0:
        return 0

    inv_scale = recip(scale)

    while True:
        sign = sample_standard_bernoulli()
        magnitude = sample_geometric_exp_fast(inv_scale) #

        if sign or magnitude != 0: #
            if sign:
                return magnitude
            else:
                return -magnitude
```

Postcondition

For any setting of the input parameter `scale` such that the given preconditions hold, `sample_discrete_laplace` either returns `Err(e)` due to a lack of system entropy, or `Ok(out)`, where `out` is distributed as $\mathcal{L}_{\mathbb{Z}}(0, scale)$.

[1] See new changes with `git diff 0be3ab3e6..6535c79 rust/src/traits/samplers/cks20/mod.rs`

2 Proof

Definition 2.1. [BV17] (Discrete Laplace). Let $\mu, \sigma \in \mathbb{R}$ with $\sigma > 0$. The discrete laplace distribution with location μ and scale s is denoted $\mathcal{L}_\mathbb{Z}(\mu, s)$. It is a probability distribution supported on the integers and defined by

$$\forall x \in \mathbb{Z} \quad P[X = x] = \frac{e^{1/s} - 1}{e^{1/s} + 1} e^{-|x|/s} \quad \text{where } X \sim \mathcal{L}_\mathbb{Z}(\mu, s)$$

Assume the preconditions are met.

Lemma 2.2. `sample_discrete_laplace` only returns `Err(e)` when there is a lack of system entropy.

Proof. By the non-negativity precondition on `scale`, the precondition on `sample_geometric_exp_fast` is met. By the definitions of `sample_geometric_exp_fast` and `sample_standard_bernoulli`, an error is only returned when there is a lack of system entropy. The only source of errors is from the invocation of these functions, therefore `sample_discrete_gaussian` only returns `Err(e)` when there is a lack of system entropy. □

We now condition on not returning an error, and establish some helpful lemmas.

Lemma 2.3. [CKS20] Let $B \sim \mathit{Bernoulli}(1/2)$ and $Y \sim \mathit{Geometric}(1 - e^{-1/s})$ for some $s > 0$. Then $P[(B, Y) \neq (\top, 0)] = \frac{1}{2}(e^{-1/s} + 1)$.

Proof.

$$\begin{aligned}
P[(B, Y) \neq (\top, 0)] &= P[B = \top, Y > 0] + P[B = \bot] && \text{by LOTP} \\
&= P[B = \top]P[Y > 0] + P[B = \bot] && \text{by independence of B, Y} \\
&= \frac{1}{2} e^{-1/s} + \frac{1}{2} \\
&= \frac{1}{2}(e^{-1/s} + 1)
\end{aligned}$$

□

Lemma 2.4. [CKS20] Given random variables $B \sim \mathit{Bernoulli}(1/2)$ and $Y \sim \mathit{Geometric}(1 - e^{-1/s})$, define $X|_{B=\top} = Y$, and $X|_{B=\bot} = -Y$. If $(B, Y) \neq (\top, 0)$, then $X \sim \mathcal{L}_\mathbb{Z}(0, \mathit{scale})$. That is, $P[X = x | (B, Y) \neq (\top, 0)] = \frac{e^{1/s} - 1}{e^{1/s} + 1} e^{-|x|/s}$ for any $x \in \mathbb{Z}$.

Proof.

$$\begin{aligned}
P[X = x | (B, Y) \neq (\top, 0)] &= \frac{P[X = x, (B, Y) \neq (\top, 0)]}{P[(B, Y) \neq (\top, 0)]} \\
&= \frac{P[X = |x|, B = \mathbb{I}[x < 0]]}{P[(B, Y) \neq (\top, 0)]} && \text{since x} = \pm y \\
&= \frac{P[X = |x|]P[B = \mathbb{I}[x < 0]]}{P[(B, Y) \neq (\top, 0)]} && \text{by independence of B, Y} \\
&= \frac{P[X = |x|]\frac{1}{2}}{\frac{1}{2}(e^{-1/s} + 1)} && \text{by 2.3} \\
&= \frac{1 - e^{-1/s}}{1 + e^{-1/s}} e^{-|x|/s} \\
&= \frac{e^{1/s} - 1}{e^{1/s} + 1} e^{-|x|/s}
\end{aligned}$$

□

Lemma 2.5. If the outcome of `sample_discrete_laplace` is `Ok(out)`, then `out` is distributed as $\mathcal{L}_{\mathbb{Z}}(0, scale)$.

Proof. In the 2.2 proof, it was established that the preconditions on `sample_geometric_exp_fast` are met. therefore `magnitude` on line 9 is distributed as $Geometric(1 - e^{-1/scale})$. Similarly, by the definition of `sample_standard_bernoulli`, `sign` is distributed according to $Bernoulli(p = 1/2)$. The branching logic from line 11 on satisfies the procedures described in 2.4. Therefore, by 2.4, `out` is distributed as $\mathcal{L}_{\mathbb{Z}}(0, scale)$. □

Proof. 1 holds by 2.2 and 2.5. □

References

[BV17] Victor Balcer and Salil P. Vadhan. Differential privacy on finite computers. *CoRR*, abs/1709.05396, 2017.

[CKS20] Clément L. Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. *CoRR*, abs/2004.00010, 2020.