Tolerance Calculation for ApproxEquals in Tests

This document explains how to calculate the tolerance needed to achieve a desired flakiness when ApproxEquals is used in Privacy on Beam tests.

ApproxEquals is used to test various things about differentially private aggregations (e.g., that the aggregation is computed correctly, that contribution bounding is done, etc.). However, we have to take into account the differential privacy noise introduced when comparing the Privacy on Beam aggregates with the desired outcome.

Throughout this document, we suppose that we wish to achieve a flakiness of 10^{-k}, i.e. ensure that a correctly-implemented mechanism passes the tests with probability $1-10^{-k}$.

Floats

Laplace Noise

When applying the Laplace mechanism with epsilon ε and ℓ_1 sensitivity s_1, we add noise X from a distribution whose CDF is given by:

$$P(X \leq x) = 1 - \frac{1}{2}\exp\left(\frac{-\varepsilon x}{s_1}\right), x \geq 0$$

The minimal tolerance t that achieves the desired flakiness satisfies $P(-t \leq X \leq t) = 1-10^{-k}$. Since X is symmetrical about 0, this is equivalent to requiring the following:

$$1 - \frac{1}{2}10^{-k} = P(X \leq t) = 1 - \frac{1}{2}\exp\left(\frac{-\varepsilon t}{s_1}\right)$$

Solving for t, we get:

$$\exp\left(\frac{\varepsilon t}{s_1}\right) = 10^k$$

$$\Leftrightarrow \frac{\varepsilon t}{s_1} = k \cdot \ln 10$$

$$\Leftrightarrow t = \frac{s_1 \cdot k \cdot \ln 10}{\varepsilon}$$

For example, suppose we use $k = 23$ in order to achieve a flakiness of 10^{-23}. With $\varepsilon = 50$ and $s_1 = 1$, we need a tolerance of $t = 1.05919$.

Gaussian Noise

When applying the Gaussian mechanism with epsilon ε, delta δ and ℓ_2 sensitivity s_2, we add noise X from a distribution whose CDF is given by:

$$P(X \leq x) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x}{\sigma\sqrt{2}}\right)\right)$$

where erf is the error function.

The minimal tolerance t that achieves the desired flakiness satisfies $P(-t \leq X \leq t) = 1-10^{-k}$. Since X is symmetrical about 0, this is equivalent to requiring the following:

$$1 - \frac{10^{-k}}{2} = P(X \leq t) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{t}{\sigma\sqrt{2}}\right)\right)$$

Solving for t, we get:

$$1-\frac{10^{-k}}{2}=\frac{1}{2}+\frac{1}{2}\mathrm{erf}\left(\frac{t}{\sigma\sqrt{2}}\right)$$
$$\Leftrightarrow\frac{1}{2}-\frac{10^{-k}}{2}=\frac{1}{2}\mathrm{erf}\left(\frac{t}{\sigma\sqrt{2}}\right)$$
$$\Leftrightarrow 1-10^{-k}=\mathrm{erf}\left(\frac{t}{\sigma\sqrt{2}}\right)$$
$$\Leftrightarrow \mathrm{erfinv}\left(1-10^{-k}\right)=\frac{t}{\sigma\sqrt{2}}$$
$$\Leftrightarrow t=\mathrm{erfinv}\left(1-10^{-k}\right)\sigma\sqrt{2}$$

where erfinv is the inverse error function.

We use math package function Erfinv to approximate erfinv and the Go DP Library's SigmaForGaussian function to compute σ with s_2, ε and δ.

Integers

The logic for floats applies to integers as well. However, the caveat is that the noise is rounded to the nearest integer for aggregations on integers.

This is not a problem when the decimal part of the tolerance needed for a given flakiness is less than 0.5 since all the noise values up to 0.5 are going to be rounded down to 0.

When the decimal part of the tolerance is greater than 0.5, however, we need to round tolerance up to the next integer since all the noise values from 0.5 up to tolerance would be rounded up.

For example, for a given flakiness, if we need a tolerance of 1.1, we can use a tolerance of 1.1 (or any value≥ 1). On the other hand, if we need a tolerance of 2.6, we would need to use a tolerance of 3.0.

That is why, we round the tolerance to the nearest integer in our tests for integers.

Multiple Partitions

When there are multiple partitions in a test, noise is applied to each partition independently. If there are n partitions and each has a flakiness of 10^{-k}, then the overall test has a probability $(1-10^{-k})^n$ of passing.

We use large values of k in our tests, so we can approximate $(1-10^{-k})^n \simeq 1-n\cdot 10^{-k}$.

For $n\leq 10$, this would mean that using per-partition flakiness of 10^{-k} would result in an overall flakiness of 10^{-k+1} across partitions.

Similarly for $10<n\leq 100$, this would mean that using per-partition flakiness of 10^{-k} would result in an overall flakiness of 10^{-k+2} across partitions.

For example, if we want to achieve an overall flakiness of 10^{-23} for a test that has 10 partitions, we need to have a per-partition flakiness of 10^{-24}.

Bounded Mean

There is no simple expression for the CDF of the mean because it is a ratio of two random variables. Therefore we need to take a different approach.

Let b_0 and b_1 represent the lower and upper bounds for the input elements, respectively. In other words, if the input element is less than lower bound it will be clamped to the lower bound, and if the element is bigger than upper bound it will be clamped to the upper bound.

We first define the normalized noisy mean, m, as $m = \frac{s}{c}$ where:

- c is the noisy count clamped to be at least 1, i.e. the larger of the noisy count and 1,
- s is the noisy normalized sum, i.e. the noisy sum of distances of the input elements from the midpoint $\bar{b} = (b_0 + b_1)/2$.

Then, the noisy mean is given by $m + \bar{b}$. For simplicity, we use m instead of the noisy mean in the remainder of this document since the noisy mean is simply m shifted by $\bar{b}$ and this doesn't affect the calculations.

The CDF of the ratio of two Laplace or two Gaussian distributions is not trivial to formulate. Instead, we use the CDF for s and c to give a lower bound for the test pass probability, or equivalently give an upper bound for the flakiness.

Consider a narrow interval around the normalized raw sum, i.e. $s^- \le s^{\text{raw}} \le s^+$ and a narrow interval around the raw count, i.e. $c^- \le c^{\text{raw}} \le c^+$. If both s and c are inside these intervals, this implies that m is inside a narrow interval around the raw normalized mean, i.e.:

$$s^- \le s \le s^+ \wedge c^- \le c \le c^+ \Rightarrow m^- \le m \le m^+$$

where m^- and m^+ are constants computed from s^-, s^+, c^- and c^+.

This means

$$P(s^- \le s \le s^+ \cap c^- \le c \le c^+) \le P(m^- \le m \le m^+)$$

Our tests check that the mean is within the narrow interval, so

$$P(\text{test pass}) = P(m^- \le m \le m^+)$$

If we pick s^-, s^+, c^- and c^+ s.t.

$$P(s^- \le s \le s^+ \cap c^- \le c \le c^+) = 1 - 10^{-k}$$

then $P(\text{test pass}) \ge 1 - 10^{-k}$.

For simplicity, we pick equal probabilities for sum and count, i.e. $P(s^- \le s \le s^+) = P(c^- \le c \le c^+) = 1 - 10^{-l}$ and solve for l.

$$1 - 10^{-k} = P(s^- \le s \le s^+ \cap c^- \le c \le c^+)$$

$$\Leftrightarrow 1 - 10^{-k} = P(s^- \le s \le s^+) \cdot P(c^- \le c \le c^+)$$

$$\Leftrightarrow 1 - 10^{-k} = (1 - 10^{-l})^2$$

$$\Leftrightarrow \sqrt{1 - 10^{-k}} = 1 - 10^{-l}$$

$$\Leftrightarrow 1 - \sqrt{1 - 10^{-k}} = 10^{-l}$$

$$\Leftrightarrow \log_{10}(1 - \sqrt{1 - 10^{-k}}) = -l$$

$$\Leftrightarrow -\log_{10}(1 - \sqrt{1 - 10^{-k}}) = l$$

So in order to get overall flakiness 10^{-k} for mean, we should choose s^-, s^+, c^- and c^+ s.t.

$$P(s^- \le s \le s^+) = P(c^- \le c \le c^+) = 1 - 10^{-l} = \sqrt{1 - 10^{-k}}$$

For simplicity, we choose s^- and s^+ to be symmetrical about the normalized raw sum s^{raw}. We can hence compute values for s^- and s^+ satisfying this equation using the helper functions we previously came up with for Laplace & Gaussian by inputting l as flakinessK.

The situation for c^- and c^+ is slightly more complicated because of the clamping for c. We compute them as we do for s^- and s^+, then clamp them to be at least 1. Note that this means c^- and c^+ are not necessarily symmetrical about the raw count c^{raw} after the clamping.

Finally, we explain how we compute m^- and m^+ from these four constants:

- if $s^- > 0$ then $m^- = \frac{s^-}{c^+}$, otherwise $m^- = \frac{s^-}{c^-}$

- if $s^+ > 0$ then $m^+ = \frac{s^+}{c^-}$, otherwise $m^+ = \frac{s^+}{c^+}$

Due to how the test helpers are set up, we don't compare m with m^- and m^+ but rather check that m is within $\max(m^+ - m^{\text{raw}}, m^{\text{raw}} - m^-)$ of m^{raw}. This gives us an even wider interval than necessary to achieve 10^{-k} flakiness at the cost of test precision.

Complementary Tolerance Calculation for checkMetricsAreNoisy

We use checkMetricsAreNoisy to ensure that Privacy on Beam is adding noise with the correct distribution. The same logic as for ApproxEquals applies here but the tolerance calculation is different because we need to have a "complementary confidence interval" that includes all values except for a narrow interval around the raw metric.

Unlike ApproxEquals, we do not round values to the nearest integer to deal with integer valued aggregations because this is a complementary tolerance.

Again, suppose we wish to achieve a flakiness of 10^{-k}, i.e. ensure that a correctly-implemented mechanism passes the tests with probability $1 - 10^{-k}$.

Laplace Noise

When applying the Laplace mechanism with epsilon ε and ℓ_1 sensitivity s_1, we add noise X from a distribution whose CDF is given by:

$$P(X \le x) = \frac{1}{2}\exp\left(\frac{\varepsilon x}{s_1}\right), x \le 0$$

The minimal tolerance t that achieves the desired flakiness satisfies $P(X \le -t \cup t \le X) = 1 - 10^{-k}$. Since X is symmetrical about 0, this is equivalent to requiring the following:

$$\frac{1}{2} - \frac{10^{-k}}{2} = P(X \le -t) = \frac{1}{2}\exp\left(\frac{-\varepsilon t}{s_1}\right)$$

Solving for t, we get:

$$\exp\left(\frac{-\varepsilon t}{s_1}\right) = 1 - 10^k$$

$$\Leftrightarrow \frac{-\varepsilon t}{s_1} = \ln\left(1 - 10^k\right)$$

$$\Leftrightarrow t = -\frac{s_1 \cdot \ln\left(1 - 10^k\right)}{\varepsilon}$$

Gaussian Noise

When applying the Gaussian mechanism with epsilon ε, delta δ and ℓ_2 sensitivity s_2, we add noise X from a distribution whose CDF is given by:

$$P(X \le x) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x}{\sigma\sqrt{2}}\right)\right)$$

where erf is the error function.

The minimal tolerance t that achieves the desired flakiness satisfies $P(X \le -t \cup t \le X) = 1 - 10^{-k}$. Since X is symmetrical about 0, this is equivalent to requiring the following:

$$\frac{1}{2} - \frac{10^{-k}}{2} = P(X \le -t) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{-t}{\sigma\sqrt{2}}\right)\right)$$

Solving for t, we get:

$$1 - 10^{-k} = 1 + \operatorname{erf}\left(\frac{-t}{\sigma\sqrt{2}}\right)$$

$$\Leftrightarrow -10^{-k} = \operatorname{erf}\left(\frac{-t}{\sigma\sqrt{2}}\right)$$

$$\Leftrightarrow \operatorname{erfinv}\left(-10^{-k}\right) = \frac{-t}{\sigma\sqrt{2}}$$

$$\Leftrightarrow t = -\operatorname{erfinv}\left(-10^{-k}\right)\sigma\sqrt{2}$$

$$\Leftrightarrow t = \operatorname{erfinv}\left(10^{-k}\right)\sigma\sqrt{2}$$

where erfinv is the inverse error function.

We use math package function Erfinv to approximate erfinv and the Go DP Library's SigmaForGaussian function to compute σ with s_2, ε and δ.

Mean

Complementary tolerance calculation for mean is non-trivial. That is why for mean, we simply check that *any* noise is added (i.e. that the noisy mean is not equal to the raw mean).