Probabilistic assertions and confidence intervals

Set a shot budget and assess sampled outcomes with an explicit confidence statement.

Undergraduate 22 min lesson Dr. Charith Jayasekara

By the end, you'll be able to

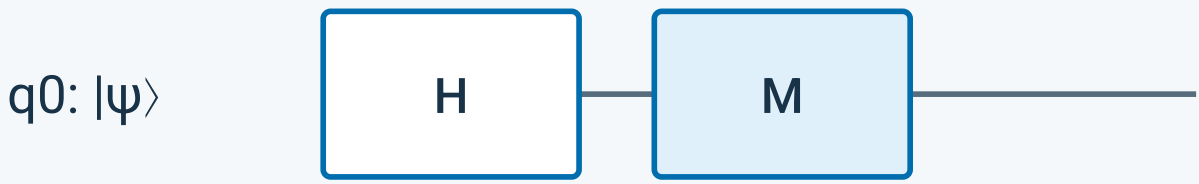
- Distinguish exact probabilities from measured frequencies.
- Calculate a conservative finite-shot tolerance.
- Separate sampling variation from systematic bias.

Begin with a clear question

Even a perfect circuit can produce different counts on repeated runs. A test that demands exactly half zeros from a balanced qubit will reject correct programs. A useful probabilistic assertion states the expected probability, sample size, allowed deviation and a bound on false rejection.

Build the reasoning

For independent Bernoulli shots, Hoeffding's inequality bounds the chance that the observed frequency differs from the true probability by at least epsilon. Solve it for epsilon after choosing a failure probability delta. This is a conservative, distribution-free bound under those independence assumptions. A confidence interval reports uncertainty in an estimate; a predeclared acceptance rule specifies when your test fails. Both need their assumptions stated.



Follow time from left to right. Qiskit statevector labels put q0 on the right; diagrams put q0 on the top wire.

Worked example

Use 4,096 samples with expected success probability one half and delta one in a million. The tolerance is about 0.0421. Draw a seeded classical Bernoulli reference to isolate the statistical assertion from circuit construction. Notebook 02

applies corresponding reasoning to Bell circuit counts and also calculates Wilson intervals.

$$\Pr(|\hat{p} - p| \geq \epsilon) \leq 2e^{-2n\epsilon^2}, \qquad \epsilon = \sqrt{\frac{\ln(2/\delta)}{2n}}.$$

```
import numpy as np
from math import log, sqrt

shots, delta, expected = 4096, 1e-6, 0.5
epsilon = sqrt(log(2 / delta) / (2 * shots))
rng = np.random.default_rng(42)
successes = int(rng.binomial(shots, expected))
frequency = successes / shots
assert abs(frequency - expected) <= epsilon
print({"shots": shots, "frequency": frequency, "tolerance": epsilon})
```

Practical lab

Choose tolerances before inspecting counts. Repeatedly changing the seed until a test passes destroys its meaning. If you test many independent conditions, budget the overall false-rejection probability, for example by assigning each test a share and applying a union bound. On hardware, drift or correlated shots may violate the simple model. Increasing shots reduces sampling uncertainty while a persistent readout bias remains.

Run the corresponding notebook from the software practice collection:

- Download Bell circuits, finite shots and honest uncertainty
- Download Quantum software tests that catch plausible wrong programs

Keep a short execution record: predicted result, observed result, environment, inputs and the check that would reject a plausible defect. Explain a failure before changing the expected value.

Chapter summary

- Predeclare the shot budget and tolerance.
- State independence and multiple-test assumptions.
- Treat sampling uncertainty and systematic bias separately.

Exercises

1. With delta fixed, how does quadrupling the shots change the Hoeffding tolerance?
2. Change one part of the worked program to create a plausible fault. State which assertion should fail and why.

Worked answers

1. The tolerance is proportional to one over the square root of n, so it halves. Reducing an error bar by ten requires roughly one hundred times as many independent shots.
2. A useful answer identifies a specific changed gate, parameter, mapping or model and links it to an independent expected behaviour. Keep the intended specification fixed while changing the implementation; save the failing case as a regression test.

Check your understanding

PAUSE & THINK

A readout process shifts the true success probability from 0.50 to 0.60. What does increasing shots do?

- [ ] Makes the estimate concentrate around 0.60
- [ ] Restores the ideal probability to 0.50
- [ ] Guarantees every run has exactly equal counts

Answer: Makes the estimate concentrate around 0.60. More independent data narrows uncertainty around the actual distribution.

Sources

IBM's bit-ordering guide, Statevector API and Operator API describe the SDK conventions and equivalence tools. Hoeffding's probability-bound paper supplies the stated independent-shot bound. The notebook bundle records its pinned environment and local execution evidence.