Turning an estimate into a fair score

The math, reasoning, edge cases, and implementation behind SurahSpot's Ayah Counts estimation mode.

Exact Figure asks, "Do you know the count?" Closest Figure asks, "How good is your estimate, and can you improve it using feedback?" That difference is why the mode uses a scoring curve instead of a simple right-or-wrong rule.

The same guess against the same Surah count always produces the same score. The system combines relative error, a short-Surah tolerance floor, a nonlinear falloff curve, best-guess scoring, and a small cost for extra tries.

1. Measure the miss

Let **a** be the actual Ayah count and **g** be the guess. The raw distance is:

```
d = |g - a|
```

The sign does not matter for scoring: five under and five over are equally close.

2. Scale the miss to the Surah

A fixed penalty per Ayah would be unfair across Surahs whose lengths range from 3 to 286. SurahSpot therefore builds a tolerance window from the target itself:

```
T = max(3, 0.5 * a)
```

Normally the scoring window is half the true count. The minimum of 3 protects very short Surahs from percentage-error behavior that would otherwise be too harsh.

3. Normalize the error

```
e = d / T
```

This turns an absolute miss into a comparable scale. e = 0 means exact. e = 0.5 means halfway through the scoring window. e >= 1 is outside the window.

4. Apply the falloff curve

```
S = round(100 * (1 - e)^1.5)
```

Exact guesses receive 100. At or beyond the tolerance boundary, the score is 0. The 1.5 exponent makes the curve stricter than a straight line: strong estimates stay valuable, while vague estimates lose value faster.

Normalized error	Linear	Current curve
0.00	100	100
0.10	90	85
0.25	75	65
0.50	50	35
0.75	25	12

Normalized error	Linear	Current curve
1.00	0	0

Why the 1.5 exponent?

A linear curve proved too generous to broad, low-information guesses. Raising the remaining accuracy to 1.5 bends the curve downward. This is a product-design choice, not a law of mathematics.

5. Higher/lower turns guessing into convergence

After a non-exact guess, the player receives direction only: Higher when the estimate is below the answer, Lower when it is above. The distance is never revealed. That gives enough information to narrow a range without solving the round.

6. Best guess carries the round

Each guess gets its own proximity score. The round keeps the best one and subtracts 5 points for each try after the first:

```
R = max(0, max(S1, S2, ..., Sn) - 5 * (n - 1))
```

Best-of avoids a bad incentive that averaging would create. A later weak guess cannot erase a strong estimate already demonstrated, while the try penalty still rewards getting there sooner.

Worked round: actual count = 78

Try	Guess	Score	Feedback
1	120	0	Lower
2	90	58	Lower
3	80	92	Lower

The best single score is 92. Three tries means two extra-try penalties, so the current round value is 92 - 10 = **82**. If try 4 is exact, the best becomes 100 and the round finishes at 100 - 15 = **85**.

Current score vs. ceiling

The current score asks, "What is my best work so far worth?" The ceiling asks, "If my next guess were perfect, what is the most I could still earn?"

```
C = max(0, 100 - 5 * triesUsed)
```

Edge cases: where the formula meets real gameplay

A fair curve can still become an unfair game if unusual inputs or repeated requests create loopholes. Closest Figure therefore treats edge handling as part of the scoring design, not as an afterthought.

Case	What SurahSpot does	Why
0, negatives, decimals, text, >300	Rejects the input before scoring.	Only plausible whole Ayah counts may consume a guess.
NaN / non-finite / impossible actual <= 0	Pure scorer safely returns 0.	Defensive math should not throw or leak NaN into the UI.
Very short Surah	Uses minimum tolerance T = 3.	A one-Ayah mistake should matter without becoming absurdly harsh.
At/outside tolerance boundary	Single-guess score is 0.	Wild guesses cost a try but never create negative points.
Skip try	Records score 0 and consumes one try.	Skipping cannot dodge uncertainty or preserve the try count.
Skip round	Ends immediately at 0 and reveals the answer.	Abandoning is different from cashing out the current best score.
Repeat the same wrong guess	Allowed, but another try penalty is incurred.	No special duplicate rule is needed; repetition is naturally costly.
Fifth try	Round closes; wrong reveals answer, exact can still score 80.	The mode always respects the five-try limit.
No guesses / all-zero scores	Round-score helper returns/clamps to 0.	Penalties never push the score negative.
Double-click / concurrent submit	Only one request on a round revision can succeed.	One user action cannot accidentally score twice.
Old/replayed round state	Rejected as stale after the revision rotates.	A saved token cannot roll the round backward.
Token moved to another browser	Rejected when it does not match the attempt cookie.	A round belongs to its attempt session.
Active round payload	Ayah count remains hidden until reveal.	The client cannot read the answer before the round ends.

The important distinction: validation vs. scoring

The API validates normal player input first: it must be an integer from 1 through 300. Separately, the pure scoring function is defensive even if called incorrectly. Non-finite inputs or an impossible actual count return 0. Keeping both layers is intentional: validation protects the game contract; defensive math protects the program.

Why a skipped try is explicitly stored as zero

Closest Figure uses best-of scoring. If a skip were simply omitted from the score history, a player could spend tries without that action being represented in the round's data. Recording 0 makes the history honest: the best score remains intact, but the try count and its 5-point cost still advance.

Why Skip Round discards the current best

Skip Round is modeled as abandoning the round, not stopping early. It therefore finishes at 0 and reveals the count. This keeps the button semantically clear and prevents it from becoming a 'bank my best score now' shortcut.

Why stale-state rejection matters

Each accepted action rotates the round revision. If two requests use the same revision, only one can mutate the attempt; the other is rejected as stale.

Implementation

The scoring math lives in pure TypeScript functions. The client can use the same scorer for a live preview, while the server remains authoritative for the actual award and round state.

```
const FALLOFF_EXPONENT = 1.5;
const TOLERANCE_RATIO = 0.5;
const MIN_TOLERANCE = 3;
export const MAX_AYAH_GUESS = 300;
export const TRY_PENALTY = 5;

export function isValidAyahGuess(value: unknown) {
  const guess = Number(value);
  return Number.isInteger(guess)
    && guess >= 1
    && guess <= MAX_AYAH_GUESS;
}

export function closestFigureScore(guess: number, actual: number) {
  if (!Number.isFinite(guess)
      || !Number.isFinite(actual)
      || actual <= 0) return 0;

  const distance = Math.abs(guess - actual);
  if (distance === 0) return 100;

  const tolerance = Math.max(
    MIN_TOLERANCE,
    actual * TOLERANCE_RATIO
  );

  const error = distance / tolerance;
  if (error >= 1) return 0;

  return Math.round(
    100 * Math.pow(1 - error, FALLOFF_EXPONENT)
  );
}

export function closestFigureRoundScore(scores: readonly number[]) {
  if (!scores.length) return 0;
  const best = Math.max(...scores);
  const penalty = TRY_PENALTY
    * (Math.min(scores.length, 5) - 1);
  return Math.max(0, best - penalty);
}
```

Server-side edge handling

On each guess, the server verifies the attempt session, validates the count, computes the authoritative score, records the per-try history, and rotates the round revision. A skipped try is appended as 0. A skipped round finishes immediately at 0. On the fifth try, the round ends and the true count is revealed.

What the tests protect

Property	Protected behavior
Exact = 100	The target remains unambiguous.
Relative scaling	Equal raw misses can mean different things on different Surah lengths.
Short-Surah floor	One-Ayah misses do not become arbitrarily harsh.
Monotonic falloff	Moving farther away can never improve the score.
Symmetry	Equal distance above/below earns the same score.
Zero floor	Wild guesses and penalties never create negative points.

Property	Protected behavior
Degenerate input	NaN / invalid actual state returns 0 instead of throwing.
Best-of scoring	A later bad guess cannot erase an earlier strong estimate.
Skipped try = 0	Skipping cannot bypass the try history.
Five-try closure	The answer is revealed when the round is exhausted.
Revision safety	Replayed or simultaneous actions cannot score twice.

The constants are design choices

0.5, 3, 1.5, and 5 are tunable parameters, not mathematical laws. Together they define the feel of Closest Figure: reward useful estimation, make fixed/random guessing weak, protect short Surahs from percentage-error weirdness, and still reward reaching the answer efficiently.

In plain English

Closest Figure asks four things: How far were you from the answer? How large is this Surah, so what does that miss really mean? How many tries did it take? And did the request itself obey the rules of the round? The curve answers the first two, the try penalty answers the third, and the validation/revision system protects the fourth.

Formula version documented here: tolerance ratio 0.5; minimum tolerance 3; falloff exponent 1.5; extra-try penalty 5; maximum five tries per round; valid player guesses are whole numbers from 1 to 300.