Estimation Score Formula

I made a game where players have to answer estimation questions using a scale. If their guess is close to the correct answer, they can get up to 1000 points. The further away they are, the fewer points they get.

For example, a question could be: What is the record for most hot dogs eaten in 10 minutes? The scale goes from 50 to 100, and the correct answer is 76 hot dogs. If a player guesses 53, how should I calculate their score?

What's a good formula to give fair points based on how close the guess is?

A good formula should reward close guesses while ensuring that further guesses receive fewer points in a fair and smooth manner.

Here's one effective approach:

Formula:

$$\text{Score} = 1000 \times \left(1 - \frac{|\text{Guess} - \text{Answer}|}{\text{Max Range} - \text{Min Range}}\right)^p$$

Explanation:

- Answer is the correct value (76 in this case).
- Guess is the player's estimate (e.g., 53).
- Min Range and Max Range define the possible answers (50 to 100).
- p is an exponent (e.g., 2) to make the scoring curve non-linear, punishing distant guesses more.

Applying the Formula to Your Example:

Given:

- $\text{Answer} = 76$
- $\text{Guess} = 53$
- $\text{Min Range} = 50$
- $\text{Max Range} = 100$
- $p = 2$ (a reasonable choice for quadratic falloff)

First, calculate the absolute difference:

$$|\text{Guess} - \text{Answer}| = |53 - 76| = 23$$

Then, normalize the error by the range:

$$\frac{23}{100 - 50} = \frac{23}{50} = 0.46$$

Apply the exponent:

$$(1 - 0.46)^2 = (0.54)^2 = 0.2916$$

Multiply by 1000 to get the final score:

$$1000 \times 0.2916 = 291.6$$

So, a guess of **53 would earn about 292 points**.

Why This Formula?

- Ensures **1000 points for a perfect guess**.
- Gradually reduces points as the guess gets further away.
- The **exponent** p makes it more punishing for large errors while still giving some points.

Would you like to tweak the parameters (like the exponent) for a different scoring curve?

Can you give me this formula as python code?

Sure! Here's a Python function that calculates the score based on your formula:

```python
def calculate_score(guess, answer, min_range, max_range, p=2):
    """
    Calculates the score based on how close the guess is to the correct answer.

    Parameters:
    guess (float): The player's guess.
    answer (float): The correct answer.
    min_range (float): The minimum possible value of the scale.
    max_range (float): The maximum possible value of the scale.
    p (float): Exponent to control the scoring curve (default is 2 for quadratic falloff).

    Returns:
    float: The calculated score (capped between 0 and 1000).
    """
    max_points = 1000
    error = abs(guess - answer) / (max_range - min_range)
    score = max_points * (1 - error) ** p
    return max(0, score)  # Ensure the score is never negative

# Example usage:
guess = 53
answer = 76
min_range = 50
max_range = 100

score = calculate_score(guess, answer, min_range, max_range)
print(f"Score: {round(score)}")  # Output should be around 292
```

Explanation:

- It calculates how far the guess is from the correct answer relative to the total range.
- The `(1 - error) ** p` part makes sure that the score decreases non-linearly.
- `max(0, score)` ensures the score never goes below zero.

You can tweak the exponent `p` to make the scoring harsher (`p > 2`) or more forgiving (`p < 2`).

Would you like to visualize how different values of `p` affect the score? 🚀