**Product Requirements Document (PRD)**

**Project Name:** Random Points Scatter and Best Fit Line

**Date:** 2025-09-27

**Prepared by:** Yair Levi

**1. Purpose**

The purpose of this project is to generate a set of random points in 2D space, evaluate multiple random linear models against these points, and visually display the line that minimizes the sum of squared errors. The final output is a scatter plot with the best-fitting line overlaid.

**2. Scope**

* Generate 1000 random points with coordinates xx and yy between 0 and 1.
* Evaluate 100 random lines of the form y=a⋅x+by = a \cdot x + b and compute the error for each line.
* Identify the line with the minimum total squared error.
* Display the points and the best-fitting line in a plot.

**3. Requirements**

**3.1 Functional Requirements**

1. **Random Points Generation**
   * Generate 1000 random points (xi,yi)(x\_i, y\_i) where xi,yi∈[0,1]x\_i, y\_i \in [0,1].
2. **Plot Points**
   * Display the points on a 2D scatter plot.
   * Points should be colored red.
3. **Random Line Generation**
   * Generate 100 random lines.
   * For each line:
     + Random slope a∈[0,tan⁡(π/2−0.0017)]a \in [0, \tan(\pi/2 - 0.0017)] with a **uniform distribution**.
     + Random intercept b∈[0,1]b \in [0, 1] with a uniform distribution.
4. **Error Calculation**
   * Compute squared error for each point and each line:

errori=(yi−(a⋅xi+b))2\text{error}\_i = (y\_i - (a \cdot x\_i + b))^2

* + Use **vectorized operations** (no loops).

1. **Error Summarization**
   * For each line, compute total error by summing errors of all points.
   * Store the errors in an array of size 100 (one per line).
2. **Find Best Line**
   * Identify the index of the line with minimal total error.
   * Retrieve the corresponding aa and bb coefficients.
3. **Plot Best-Fit Line**
   * Draw the line y=a⋅x+by = a \cdot x + b over the scatter plot.
4. **Graph Title**
   * The scatter plot with the best-fit line should have the title: "Scatter plot of points".

**3.2 Non-Functional Requirements**

* **Performance:** Use vectorized calculations to efficiently compute errors without loops.
* **Portability:** The code should run on Python 3.x using standard libraries like NumPy and Matplotlib.
* **Maintainability:** Code should be modular, with functions for generating points, generating lines, computing errors, and plotting.
* **Reproducibility:** Optionally allow setting a random seed to reproduce results.

**3.3 Constraints**

* Avoid slopes that reach vertical (π/2\pi/2) to prevent infinite slopes.
* Use vectorized operations instead of explicit loops for performance.
* Limit the number of points to 1000 and number of lines to 100.

**4. Deliverables**

1. Python script or Jupyter notebook implementing the functionality.
2. Scatter plot with red points and the best-fit line overlaid.
3. Array containing the total errors for each of the 100 lines.

**5. Acceptance Criteria**

* 1000 points are generated correctly within [0,1].
* 100 random lines are generated with slopes and intercepts in the specified ranges.
* Total squared errors are calculated without using loops.
* The minimal-error line is correctly identified.
* The final plot shows red points and the best-fitting line with the title "Scatter plot of points".